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

Навчально-науковий інститут телекомунікаційних систем

Кафедра телекомунікацій

«На правах рукопису»
УДК _____

«До захисту допущено»
Завідувач кафедри
_____ Сергій КРАВЧУК
«___»_____2025 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою «Інженерія та програмування інфокомунікацій»
зі спеціальності 172 «Електронні комунікації та радіотехніка»

на тему: «Використання технологій штучного інтелекту для проведення досліджень в сфері телекомунікацій»

Виконав:
студент ІІ курсу, групи ЦК-41мп
Міндрул Дмитро Сергійович _____

Керівник:
Доцент кафедри ТК НН ІТС, к.т.н., с.н.с.,
Міночкін Дмитро Анатолійович _____

Рецензент:
Доцент кафедри ІТТ НН ІТС, к.т.н., доцент
Новогрудська Ріна Леонідівна _____

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

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий інститут телекомунікаційних систем
Кафедра телекомунікацій

Рівень вищої освіти – другий (магістерський)

Спеціальність – 172 «Телекомунікації та радіотехніка»

Освітньо-професійна програма «Інженерія та програмування інфокомунікацій»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Сергій КРАВЧУК
«___»_____2024 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Міндрулу Дмитру Сергійовичу

1. Тема дисертації «Використання технологій штучного інтелекту для проведення досліджень в сфері телекомунікацій», науковий керівник дисертації Міночкін Дмитро Анатолійович, к.т.н., с.н.с., затверджені наказом по університету від «03» листопада 2025 р. № 4772-с.

2. Термін подання студентом дисертації: 15.12.2025 р.

3. Об’єкт дослідження: методологія та процеси науково-дослідної роботи в галузі телекомунікацій, зокрема пошук, аналіз та синтез науково-технічної інформації.

4. Предмет дослідження: архітектура, функціональність та ефективність спеціалізованого ШІ-агента для автоматизації наукового пошуку, аналізу та синтезу інформації в предметній області телекомунікацій.

5. Перелік завдань, які потрібно розробити:

1) Провести системний аналіз сучасних методів та інструментів автоматизації наукових досліджень. Дослідити існуючі підходи до використання технологій штучного інтелекту для наукового пошуку, аналізу та синтезу знань, зокрема в технічних галузях. Визначити їхні переваги, обмеження та актуальні виклики;
2) Обґрунтувати архітектурні рішення та вибрати технологічний стек для побудови спеціалізованого ШІ-агента. На основі аналізу вимог до адаптивності, роботи з галузевими даними та інтеграції зовнішніх джерел обґрунтувати вибір фреймворку та супутніх технологій для створення функціональної системи

3) Спроектувати, адаптувати та налаштувати ШІ-агента для вирішення дослідницьких завдань у сфері телекомунікацій. Реалізувати кастомізацію агента для роботи з професійною термінологією, авторитетними джерелами та формування структурованих технічних звітів. Практично продемонструвати його роботу на комплексному прикладі;
4) Проаналізувати ефективність, обмеження та практичну цінність розробленої системи. Узагальнити результати експерименту, оцінити економічну доцільність, виявити технологічні та методологічні обмеження. Сформулювати висновки щодо можливостей вдосконалення системи та її потенційного впровадження як стартап-проєкт;

6. Орієнтовний перелік графічного (ілюстративного) матеріалу:

1) Тема, актуальність, мета та задачі магістерської дисертації;
2) Узагальнена блок-схема архітектури розробленого локального ШІ-серверу з агентом;
3) Приклад робочого процесу (workflow) ШІ-агента при виконанні типового дослідницького завдання;
4) Скріншоти або фрагменти результатів роботи агента;
5) Загальні висновки та перспективи подальшого розвитку системи.

7. Дата видачі завдання “02” вересня 2024 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Огляд наукової та технічної літератури щодо сучасних ШІ-агентів на основі LLM та напрямів автоматизації в ТК.	02.09.2024 – 25.10.2024	Виконано
2	Дослідження архітектурних підходів та аналіз існуючих платформ і інструментів для розробки ШІ-агентів.	28.10.2024 – 20.12.2024	Виконано
3	Визначення задачі аналізу телекомунікаційних даних для автоматизації та збирання набору даних.	07.01.2025 – 04.02.2025	Виконано
4	Вибір оптимальної LLM-платформи та інструментів для реалізації прототипу.	05.02.2025 – 03.03.2025	Виконано
5	Проектування архітектури та логіки роботи прототипу ШІ-агента.	04.03.2025 – 25.04.2025	Виконано
6	Розробка та впровадження прототипу, його налаштування та тестування.	28.04.2025 – 05.09.2025	Виконано
7	Проведення всебічного тестування ефективності прототипу та порівняльний аналіз з традиційними методами.	08.09.2025 – 19.10.2025	Виконано

Студент Дмитро МІНДРУЛ

Науковий керівник дисертації Дмитро МІНОЧКІН

РЕФЕРАТ

Пояснювальна записка до магістерської дисертації на тему «Використання технологій штучного інтелекту для проведення досліджень в сфері телекомунікацій» містить 113 сторінок, 14 рисунків, 10 таблиць, 1 додаток, список використаних джерел (18 джерел).

Актуальність теми. Сучасна телекомунікаційна галузь характеризується експоненційним зростанням обсягів науково-технічної інформації, що робить традиційні ручні методи наукового пошуку та аналізу неефективними. Це створює нагальну потребу в автоматизації дослідницьких процесів за допомогою технологій штучного інтелекту, зокрема автономних ШІ-агентів.

Зв'язок роботи з науковими програмами, планами, темами. Дисертація виконується як індивідуальна дослідницька робота і не має прямого фінансування в рамках конкретних державних чи галузевих програм. Її результати корелюють із загальними стратегічними напрямами розвитку інформаційно-комунікаційних технологій та ШІ.

Мета й завдання дослідження. Метою роботи є розробка методології, практична реалізація та оцінка ефективності спеціалізованого автономного ШІ-агента для автоматизації науково-дослідної діяльності у сфері телекомунікацій. Для цього вирішувалися завдання: системний аналіз сучасних методів автоматизації; обґрунтування архітектури та вибір технологічного стеку; проектування, адаптація та налаштування агента; аналіз ефективності та розробка бізнес-моделі.

Об'єкт дослідження. Об'єктом дослідження виступає методологія та процеси науково-дослідної роботи в галузі телекомунікацій.

Предмет дослідження. Предметом дослідження є архітектура, функціональність та ефективність спеціалізованого автономного ШІ-агента для автоматизації наукового пошуку, аналізу та синтезу інформації в телекомунікаційній галузі.

Методи дослідження. Застосовано комплекс методів: системний та порівняльний аналіз наукових джерел; методи програмної інженерії (модульне проектування, інтеграція компонентів); експериментальна перевірка; бізнес-

моделювання.

Наукова новизна одержаних результатів. Новизна полягає в комплексному підході до створення автономного ШІ-агента, спеціалізованого для досліджень у телекомунікаціях. Вперше запропоновано та реалізовано архітектурне рішення, що поєднує циклічну логіку «Scoping – Research – Synthesis», глибоку галузеву адаптацію та механізми пошуку в реальному часі. Отримано нові емпіричні дані щодо економічної ефективності подібних систем.

Практичне значення одержаних результатів. Результатом роботи є функціональний прототип ШІ-агента, здатний значно прискорювати та здешевлювати етапи пошуку, аналізу та синтезу інформації. Отриманий технологічний результат трансформовано в детальну бізнес-модель стартап-проєкту «Telecom Research Assistant as a Service» (TRaaS), що обґрунтовує його комерційний потенціал для ринку B2B-послуг.

Апробація результатів дисертації. На момент завершення роботи над дисертацією її результати ще не були апробовані на наукових конференціях або в публікаціях.

Публікації. За темою дисертації на момент її захисту наукові публікації відсутні. Основні результати вперше викладені в даній роботі.

Ключові слова: штучний інтелект, ШІ-агент, автоматизація досліджень, телекомунікації, Open Deep Research, LangGraph, RAG, SaaS.

ABSTRACT

Explanatory note to the master's thesis on the topic "Use of artificial intelligence technologies for conducting research in the field of telecommunications" contains 113 pages, 14 figures, 10 tables, 1 appendix, list of sources used (18 sources).

Relevance of the topic. The modern telecommunications industry is characterized by an exponential growth in the volume of scientific and technical information, which makes traditional manual methods of scientific search and analysis ineffective. This creates an urgent need to automate research processes using artificial intelligence technologies, in particular autonomous AI agents.

Connection of work with scientific programs, plans, topics. The dissertation is carried out as an individual research work and does not have direct funding within the framework of specific state or industry programs. Its results correlate with the general strategic directions of development of information and communication technologies and AI.

Purpose and objectives of the research. The purpose of the work is to develop a methodology, practical implementation and assessment of the effectiveness of a specialized autonomous AI agent for automating research and development activities in the telecommunications sector. For this purpose, the following tasks were solved: system analysis of modern automation methods; justification of the architecture and selection of a technological stack; design, adaptation and configuration of the agent; analysis of effectiveness and development of a business model.

Object of research. The object of research is the methodology and processes of research and development in the telecommunications sector.

Subject of research. The subject of research is the architecture, functionality and effectiveness of a specialized autonomous AI agent for automating scientific search, analysis and synthesis of information in the telecommunications sector.

Research methods. A set of methods was applied: system and comparative analysis of scientific sources; software engineering methods (modular design, component integration); experimental verification; business modeling.

Scientific novelty of the results obtained. The novelty lies in the integrated

approach to creating an autonomous AI agent specialized for telecommunications research. For the first time, an architectural solution was proposed and implemented that combines the cyclical logic of “Scoping – Research – Synthesis”, deep industry adaptation and real-time search mechanisms. New empirical data on the economic efficiency of such systems were obtained.

Practical significance of the results obtained. The result of the work is a functional prototype of an AI agent, capable of significantly accelerating and reducing the cost of the stages of information search, analysis and synthesis. The obtained technological result was transformed into a detailed business model of the startup project “Telecom Research Assistant as a Service” (TRaaS), which substantiates its commercial potential for the B2B services market.

Approbation of the results of the dissertation. At the time of completion of the work on the dissertation, its results had not yet been approved at scientific conferences or in publications.

Publications. There are no scientific publications on the topic of the dissertation at the time of its defense. The main results are presented for the first time in this work.

Keywords: artificial intelligence, AI agent, research automation, telecommunications, Open Deep Research, LangGraph, RAG, SaaS.

ПЕРЕЛІК СКОРОЧЕНЬ

ШІ	Штучний Інтелект
5G/6G	Пп’яте / шосте покоління мобільних мереж
ACM	Association for Computing Machinery (Асоціація обчислювальної техніки)
API	Application Programming Interface (інтерфейс прикладного програмування)
BERT	Bidirectional Encoder Representations from Transformers (двонаправлені кодувальні представлення трансформерів)
CNN	Convolutional Neural Network (згорткова нейронна мережа)
CSS	Cascading Style Sheets (каскадні таблиці стилів)
HNSW	Hierarchical Navigable Small Worlds (ієрархічні навігаційні малі світи)
HTML	HyperText Markup Language (мова гіпертекстової розмітки)
IEEE	Institute of Electrical and Electronics Engineers (Інститут інженерів з електротехніки та електроніки)
IoT	Internet of Things (Інтернет речей)
JSON	JavaScript Object Notation (текстовий формат обміну даними)
LLM	Large Language Model (велика мовна модель)
LSTM	Long Short-Term Memory (довгострокова короткочасна пам’ять)
MIMO	Multiple Input Multiple Output (багатовхідний багатовихідний метод)
MCP	Model Context Protocol (протокол контексту

	моделі)
NFV	Network Functions Virtualization (віртуалізація мережевих функцій)
NLP	Natural Language Processing (обробка природної мови)
O-RAN	Open Radio Access Network (відкрита радіодоступна мережа)
ODR	Open Deep Research (відкриті глибокі дослідження)
RAG	Retrieval-Augmented Generation (генерація, посилена пошуком)
RFC	Request for Comments (документ із визначення стандартів Інтернету)
RIC	Radio Access Network Intelligent Controller (інтелектуальний контролер радіодоступу)
RNN	Recurrent Neural Network (рекурентна нейронна мережа)
RAO	Reasoning-Action-Observation (мислення-дія-спостереження)
SDN	Software-Defined Networking (програмно-визначене мережування)
XML	eXtensible Markup Language (розширювана мова розмітки)

ЗМІСТ

ВСТУП 12

РОЗДІЛ 1 15

СИСТЕМНИЙ АНАЛІЗ МЕТОДІВ ТА ІНСТРУМЕНТІВ АВТОМАТИЗАЦІЇ НАУКОВИХ ДОСЛІДЖЕНЬ 15

1.1 Сучасні виклики науково-дослідної роботи в галузі телекомунікацій 15
1.1.1 Обсяг та динамічність наукової інформації 15
1.1.2 Потреба в міждисциплінарному аналізі 16
1.1.3 Вимоги до академічної строгості та відтворюваності результатів 17
1.2 Традиційні методи наукового пошуку та аналізу літератури 19
1.2.1 Пошукові системи та академічні бази даних 19
1.2.2 Ручний аналіз та систематизація джерел 20
1.2.3 Обмеження традиційних підходів 21
1.3 Сучасні технології ШІ в науковій діяльності 22
1.3.1 Системи на основі LLM загального призначення 23
1.3.2 Спеціалізовані наукові асистенти та інструменти 24
1.3.3 Підходи до автоматизації наукового пошуку та "research agents" 25
1.4 Сфери застосування технологій ШІ в наукових дослідженнях 27
1.4.1 Автоматизація систематичного огляду літератури 27
1.4.2 Аналіз та візуалізація складних даних 28
1.4.3 Написання та форматування наукових текстів 29
1.4.4 Специфіка застосування в галузі телекомунікацій 30
Висновки 31

РОЗДІЛ 2 33

АРХІТЕКТУРИ ТА ТЕХНОЛОГІЇ ПОБУДОВИ СПЕЦІАЛІЗОВАНИХ ШІ-АГЕНТІВ ДЛЯ НАУКОВОГО АНАЛІЗУ 33

2.1 Архітектурні підходи до створення автономних ШІ-агентів 33
2.1.1 Модель "Мислення-Дії-Спостереження" 33
2.1.2 Фреймворки для побудови агентів 35
2.2 Технології семантичного пошуку та роботи з науковими документами 39
2.2.1 Векторні бази даних та пошук за подібністю 39
2.2.2 Техніки RAG для забезпечення релевантності відповідей 41

2.2.3 Обробка різноформатних наукових джерел ... 42
2.3 Інструменти для доступу до зовнішніх даних та API ... 44
2.3.1 Інтеграція з академічними API ... 44
2.3.2 Можливості веб-краулінгу для пошуку актуальних публікацій ... 47
2.4 Існуючі open-source рішення для наукових досліджень ... 49
2.4.1 Огляд та порівняння проєктів типу ODR, GPT Researcher, ResearchGPT 49
2.4.2 Критерії вибору базової архітектури для власної розробки ... 53
Висновки ... 56
РОЗДІЛ 3 ... 58
РОЗРОБКА ТА НАЛАШТУВАННЯ ДОСЛІДНИЦЬКОГО ШІ-АСИСТЕНТА ... 58
3.1 Архітектура та компоненти системи ... 58
3.2 Адаптація системи під предметну область ... 70
3.3 Інтеграція та запуск системи ... 72
3.3.1 Локальне розгортання ... 72
3.3.2 Налаштування доступу до LLM через OpenRouter ... 74
3.4 Демонстрація роботи системи ... 78
3.4.1 Приклад запиту та аналіз роботи агента ... 78
3.4.2 Оцінка результатів та обговорення обмежень ... 83
Висновки ... 85
РОЗДІЛ 4 ... 88
СТАРТАП-ПРОЄКТ ... 88
4.1 Концептуальні засади стартап-проєкту ... 88
4.2 Опис продукту ... 89
4.3 Аналіз ринкових можливостей та цільових сегментів ... 91
4.4 Конкурентний аналіз ... 94
4.5 Бізнес-модель та стратегія виходу на ринок ... 97
4.6 Практична реалізація ... 100
Висновки ... 102
ЗАГАЛЬНІ ВИСНОВКИ ПО РОБОТІ ... 105
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ... 107
ДОДАТКИ ... 109
Додаток А ... 109

ВСТУП

Актуальність. Сучасна телекомунікаційна галузь, яка стрімко розвивається в контексті технологій 5G/6G, IoT та хмарних сервісів, характеризується експоненційним зростанням обсягів науково-технічної інформації. Традиційні ручні методи наукового пошуку та аналізу літератури стають неефективними, призводячи до інформаційного перевантаження дослідників та уповільнення інноваційних циклів. Це створює нагальну потребу в автоматизації дослідницьких процесів. Технології штучного інтелекту, зокрема автономні ШІ-агенти, пропонують трансформаційний підхід, здатний імітувати наукове мислення для семантичного пошуку, глибинного аналізу та синтезу знань. Актуальність дослідження посилюється специфікою галузі, що вимагає високої точності роботи з професійною термінологією та технічними стандартами, а також потребою в створенні контрольованих, адаптивних та економічно ефективних інструментів, які посилюють, а не заміщають експертну думку.

Зв'язок роботи з науковими програмами, планами, темами. Дана магістерська дисертація виконується як індивідуальна дослідницька робота, спрямована на розв'язання актуальної прикладної проблематики автоматизації наукової діяльності. Робота не має прямого фінансування в рамках конкретних державних чи галузевих науково-технічних програм.

Метою даної магістерської роботи є розробка методології, практична реалізація та комплексна оцінка спеціалізованого автономного ШІ-агента для автоматизації науково-дослідної діяльності у сфері телекомунікацій. Кінцевим результатом має стати не лише функціональний прототип інтелектуальної системи, але й обґрунтування її ефективності та практичної цінності, включаючи розроблену концепцію комерційного впровадження.

Для досягнення поставленої мети в роботі вирішуються наступні **завдання**:

1) провести системний аналіз сучасних методів та інструментів автоматизації наукових досліджень, зокрема в контексті телекомунікацій;
2) обґрунтувати архітектурні рішення та вибрати технологічний стек для побудови спеціалізованого ШІ-агента;

3) спроектувати, адаптувати та налаштувати ШІ-агента для вирішення дослідницьких завдань у сфері телекомунікацій, продемонструвавши його працездатність;
4) проаналізувати ефективність, обмеження та практичну цінність розробленої системи, трансформувавши технологічний результат у бізнес-модель стартап-проєкту.

Об'єктом дослідження виступає методологія та процеси науково-дослідної роботи в галузі телекомунікацій, зокрема його ключові етапи, пов'язані з інформаційним пошуком, критичним аналізом науково-технічної літератури та синтезом нових знань.

Предметом дослідження є архітектура, функціональність та ефективність спеціалізованого автономного ШІ-агента для автоматизації наукового пошуку, аналізу та синтезу інформації в предметній області телекомунікацій.

Методи дослідження. Методологічна база дослідження ґрунтується на комплексному підході. Теоретична частина побудована на системному та порівняльному аналізі наукових джерел та існуючих open-source рішень для побудови ШІ-агентів. Практична частина реалізована через методи програмної інженерії, включаючи модульне проектування та інтеграцію готових компонентів (фреймворк Open Deep Research, LangGraph, векторні бази, API Tavily та OpenRouter). Емпірична перевірка ефективності проведена шляхом експерименту з виконання системою контрольних дослідницьких завдань та подальшого експертного аналізу результатів. Окремі економічні та комерційні аспекти досліджені за допомогою методу бізнес-моделювання.

Наукова новизна одержаних результатів. Наукова новизна роботи полягає у комплексному підході до створення автономного ШІ-агента, спеціалізованого саме для дослідницьких задач у технічній галузі телекомунікацій. Вперше запропоновано та реалізовано архітектурне рішення, що поєднує циклічну логіку «Scoping – Research – Synthesis», глибоку галузеву адаптацію та механізми пошуку в реальному часі для роботи з актуальними джерелами. Окремі аспекти новизни включають розроблену методику трансформації академічного прототипу на базі

open-source фреймворку в детально обґрунтовану сервісну бізнес-модель (TRaaS), а також отримання нових емпіричних даних щодо економічної ефективності та граничних можливостей подібних систем для галузевого аналізу.

Практичне значення одержаних результатів. Практичним результатом роботи є функціонуючий прототип спеціалізованого автономного ШІ-агента для автоматизації досліджень у телекомунікаційній галузі, розроблений на базі фреймворку Open Deep Research. Система практично підтвердила свою здатність значно прискорювати та здешевлювати етапи пошуку, аналізу та синтезу інформації. Отриманий технологічний результат трансформовано в детально розроблену бізнес-модель стартап-проєкту «Telecom Research Assistant as a Service» (TRaaS), що обґрунтовує його комерційний потенціал для ринку B2B-послуг. Робота також містить систематизований аналіз переваг, обмежень та рекомендацій щодо вдосконалення подібних систем, що становить цінну основу для подальших досліджень та практичного впровадження.

Апробація результатів дисертації. Результати, отримані в ході даного дослідження, на момент завершення роботи над дисертацією ще не були апробовані на наукових конференціях або в публікаціях.

Публікації. За темою магістерської дисертації на момент її захисту публікації відсутні. Основні наукові результати та висновки вперше викладені в даній роботі.

РОЗДІЛ 1

СИСТЕМНИЙ АНАЛІЗ МЕТОДІВ ТА ІНСТРУМЕНТІВ АВТОМАТИЗАЦІЇ НАУКОВИХ ДОСЛІДЖЕНЬ

1.1 Сучасні виклики науково-дослідної роботи в галузі телекомунікацій

Сфера телекомунікацій переживає період безпрецедентної трансформації, зумовленої стрімким технологічним прогресом, конвергенцією дисциплін та глобалізацією наукового співробітництва. Ця динаміка формує новий, складний контекст для науково-дослідної діяльності, висуваючи низку системних викликів, які істотно впливають на якість, ефективність та значущість одержуваних результатів. Учасники наукового процесу стикаються не лише з технічними складнощами, але й з організаційними, методологічними та комунікативними бар’єрами. Аналіз цих викликів є фундаментальним кроком для розуміння потреби в розробці та впровадженні новітніх методів і інструментів автоматизації наукових досліджень. Систематизація основних проблем дозволяє виділити три взаємопов’язані ключові напрями: надзвичайно швидке зростання та старіння інформаційних масивів, необхідність інтеграції знань із суміжних дисциплін, а також підвищені вимоги до академічної строгості та можливості відтворення результатів. Кожен із цих аспектів потребує детального розгляду, оскільки саме вони визначають сучасні рамки проведення наукових пошуків у галузі телекомунікацій.

1.1.1 Обсяг та динамічність наукової інформації

Одним із найбільш помітних викликів сучасної науки в галузі телекомунікацій є експоненційне зростання обсягу наукової інформації та її надзвичайно висока динаміка [11]. Це зростання безпосередньо пов’язане з розвитком таких напрямів, як інтернет речей, хмарні обчислення, мобільні мережі п’ятого та шостого покоління (5G/6G), що спричиняє лавиноподібне збільшення кількості публікацій, патентів, технічних звітів і матеріалів конференцій. Провідні міжнародні бази даних, такі як IEEE Xplore, ACM Digital Library або Scopus, щорічно поповнюються тисячами нових робіт, що створює критичне навантаження

на дослідника на етапі інформаційного пошуку. Постає завдання не лише знайти відповідні джерела, але й ефективно відфільтрувати їх, оцінити релевантність та синтезувати актуальні знання з-поміж масиву часто суперечливих або дубльованих даних. При цьому актуальність інформації втрачається надзвичайно швидко: нові технологічні рішення, оновлення стандартів зв'язку, зміни в протоколах безпеки можуть кардинально змінити технічні та концептуальні засади впродовж декількох років, а інколи й місяців.

Цю проблему посилює фрагментованість інформаційних джерел, коли важливі дані розпорошені між численними спеціалізованими журналами, корпоративними звітами, препринтами у відкритих архівах (на кшталт arXiv) та закритими промисловими репозитаріями. Така роздробленість ускладнює формування цілісної картини досліджуваного явища та вимагає значних витрат часу на консолідацію знань. Для українських науковців додатковим складним аспектом є необхідність інтеграції в єдиний аналітичний простір публікацій англійською мовою, що домінують у глобальному науковому дискурсі, та досліджень, представлених українською в національних виданнях і звітах. Це створює мовні та регіональні бар'єри, які можуть обмежувати видимість важливих локальних результатів на міжнародному рівні та, навпаки, ускладнювати повноцінне врахування світових тенденцій в україномовному науковому середовищі. Отже, інформаційне середовище дослідника характеризується не лише надлишком, але й постійною мінливістю та гетерогенністю, що вимагає нових підходів до управління знаннями.

1.1.2 Потреба в міждисциплінарному аналізі

Сучасні телекомунікаційні системи перестали бути суто технічними об'єктами; вони є складними соціотехнічними комплексами, розробка та дослідження яких вимагають інтеграції знань з різних галузей. Поряд із традиційними для галузі електротехнікою та комп'ютерними науками, все більшого значення набувають економічне моделювання (для оцінки життєвого циклу та бізнес-моделей), правові аспекти (регулювання, інтелектуальна власність), соціологія та психологія користувача (адаптація інтерфейсів, аналіз

поведінки), а також науки про дані та штучний інтелект. Однак така міждисциплінарність породжує низку глибинних проблем. Насамперед, це епістемологічні бар'єри, коли різні науки оперують несхожими парадигмами, базовими поняттями та критеріями істинності. Наприклад, підходи до моделювання мережевих процесів з позицій теоретичної інформатики можуть суттєво відрізнятися від підходів, прийнятих у прикладній соціології для оцінки соціального впливу цих технологій, що ускладнює спільну інтерпретацію результатів та формування єдиної теоретичної бази.

Методологічна несумісність є другим серйозним викликом. Поєднання кількісних методів (математичне моделювання, статистичний аналіз) з якісними (глибинні інтерв'ю, експертні опитування) вимагає від дослідницької групи або окремого науковця надзвичайно широкого спектру компетенцій. Різні стандарти збору даних, вимоги до їх валідації та аналізу ускруднюють агрегацію результатів у цілісні висновки. Ці труднощі часто посилюються організаційними структурами, такими як «дисциплінарні силоси» університетів та наукових інститутів, де фінансування, публікаційна активність і кар'єрне зростання традиційно орієнтовані на вузьку спеціалізацію. Міждисциплінарні проекти часто опиняються в положенні маргінальних, їм складно отримати підтримку в рамках галузевих грантів або опублікувати результати у виданнях з високим імпакт-фактором, які, як правило, сфокусовані на конкретних дисциплінах. Крім того, відсутність загальновизнаних критеріїв оцінки якості та значущості міждисциплінарних робіт призводить до того, що вони можуть ігноруватися або недооцінюватися академічною спільнотою.

1.1.3 Вимоги до академічної строгості та відтворюваності результатів

Принципи наукової доброчесності, що включають прозорість, строгість методів та можливість відтворення результатів, набувають особливої ваги у дослідженнях телекомунікацій, водночас реалізація цих принципів зустрічає численні перепони [13]. Критичною проблемою є доступність якісних даних. Багато прикладних досліджень базуються на операторських даних трафіку, конфігураціях мереж або характеристиках обладнання, які є комерційною таємницею або захищені з міркувань кібербезпеки. Ця обмеженість доступу до

первинних даних унеможливлює незалежне підтвердження опублікованих висновків, їхню верифікацію та повторне використання іншими науковими групами для порівняльних аналізів або розвитку ідей. Ситуацію ускладнює гетерогенність телекомунікаційних платформ і середовищ: експеримент, проведений на обладнанні певного вендора з конкретними налаштуваннями протоколів, може бути практично неможливо точно відтворити в іншій лабораторній чи реальній середі.

Відсутність уніфікованих, загальноприйнятих стандартів для опису методології експериментів, вибору метрик оцінки та процедур обробки даних є ще одним серйозним бар’єром для відтворюваності. Навіть за наявності відкритого коду або опису алгоритму, відмінності в деталях реалізації можуть призвести до суттєво розбіжних результатів. Часто через обмеження обсягу статті або політику видавця автори не наводять усіх необхідних технічних деталей, що робить опубліковану роботу скоріше демонстрацією концепції, ніж інструкцією для точного повторення. Швидка еволюція технологій ще більше девальвує зусилля з детальної документації: обладнання оновлюється, протоколи модифікуються, і конфігурація, в якій був отриманий результат, може стати недоступною вже за кілька років.

Публікаційна політика, орієнтована переважно на представлення нових, інноваційних позитивних результатів, часто маргіналізує важливість публікації негативних результатів або статей, присвячених саме відтворенню та верифікації попередніх досліджень. Це створює перекос у науковому дискурсі та обмежує можливості для кумулятивного накопичення знань. Фінансові та ресурсні обмеження також відіграють ключову роль, оскільки вартість сучасного телекомунікаційного обладнання, ліцензій на спеціалізоване програмне забезпечення та потужних обчислювальних ресурсів може бути непідйомною для наукових груп з країн, що розвиваються, або для малозабезпечених лабораторій, ще більше поглиблюючи нерівність у можливостях проведення якісних, відтворюваних досліджень. Для українських дослідників додається специфіка регіональних нормативних баз та особливостей національної інфраструктури, що

може обмежувати можливості прямого порівняння їхніх результатів із міжнародними аналогами та ускладнювати інтеграцію в глобальний науковий процес на умовах паритету.

1.2 Традиційні методи наукового пошуку та аналізу літератури

Формування наукової позиції, обґрунтування актуальності дослідження та побудова методології в сучасній телекомунікаційній науці нерозривно пов’язані з якісним аналізом наявного масиву публікацій. Традиційні методи пошуку та аналізу літератури, що складалися протягом десятиліть, становлять основу академічної культури. Вони спрямовані на систематичне виявлення, оцінку та синтез наукових знань, що дозволяє дослідникам визначати контекст своєї роботи, ідентифікувати прогалини у знаннях та будувати на фундаменті попередніх досягнень. Ці методи ґрунтуються на комбінації використання спеціалізованих інформаційних систем та інтелектуального ручного аналізу, що вимагає від науковця критичного мислення, уважності та глибокого розуміння предметної галузі. Однак у світлі експоненційного зростання обсягу інформації та ускладнення дослідницьких питань класичні підходи демонструють низку системних обмежень, вивчення яких є необхідним для розуміння еволюції методології наукового пошуку.

1.2.1 Пошукові системи та академічні бази даних

Ключовим інструментом будь-якого наукового дослідження є академічні бази даних та пошукові системи, які виступають у ролі шлюзів до світового корпусу знань. У галузі телекомунікацій дослідники традиційно оперують кількома основними платформами, кожна з яких має свої специфічні характеристики, переваги та недоліки. Провідною спеціалізованою базою є IEEE Xplore Digital Library, яка концентрує в собі знання, згенеровані під егідою IEEE та IET. Цей ресурс пропонує безпрецедентну глибину охоплення технічної літератури, включаючи журнальні статті, матеріали конференцій, стандарти та книги, що робить його незамінним для відстеження суворо рецензованих інженерних робіт. Висока довіра до якості контенту та розширені можливості пошуку за детальними метаданими є його ключовими перевагами. Проте його фокус на публікаціях конкретних спільнот одночасно обмежує міждисциплінарність, а модель доступу

на основі підписки створює бар’єри для дослідників без інституційної підтримки.

На противагу спеціалізації IEEE Xplore, такі бази даних, як Scopus та Web of Science, пропонують широкий міждисциплінарний охоплення. Scopus, розроблений Elsevier, агрегує десятки тисяч рецензованих журналів, забезпечуючи потужні інструменти для аналізу цитувань, відстеження наукового впливу авторів та установ. Його сила полягає в можливості відобразити міжгалузеві зв’язки та глобальні тренди. Однак його зміст також залежить від підписки, а відбір матеріалів часто орієнтований на міжнародні видання з високим імпакт-фактором, що може маргіналізувати важливі регіональні, національні чи україномовні публікації, які не входять до основного потоку. Google Scholar виступає універсальним та вільним доповненням до комерційних баз. Його алгоритми агрегують контент з різноманітних джерел, включаючи наукові репозитарії, сайти університетів та особисті профілі дослідників, що забезпечує широту охоплення, особливо в частині препринтів та «сірої літератури». Незважаючи на це, його основними недоліками є відсутність чіткого контролю якості, можливе дублювання записів та недостатньо структуровані метадані, що ускладнює точний і відтворюваний пошук для систематичних оглядів. Таким чином, дослідник змушений комбінувати кілька систем, усвідомлюючи сліпі зони кожної, що значно ускладнює та подовжує процес формування вичерпної бібліографічної бази.

1.2.2 Ручний аналіз та систематизація джерел

Після ідентифікації потенційно релевантних джерел настає найбільш трудомістка та інтелектуально навантажена фаза — їх ручний аналіз і систематизація. Цей процес розпочинається з ретельного вибору та ітеративного уточнення ключових слів і пошукових стратегій. Дослідник формує та адаптує словниковий запас, що включає синоніми, специфічні технічні терміни, назви стандартів (наприклад, 3GPP, RFC) та коди технологій (наприклад, MIMO, NFV), балансуючи між чутливістю пошуку (повнотою) та специфічністю (точністю). Подальша робота часто ґрунтується на техніці трекінгу цитувань, де через аналіз бібліографічних списків ключових статей («зворотний пошук») або виявлення новіших робіт, які на них посилаються («прямий пошук»), будується карта

наукових зв’язків. Цей метод дозволяє простежити еволюцію ідей, однак його ефективність безпосередньо залежить від якості вихідних джерел та можливостей інструментів аналізу цитувань у використовуваних базах даних.

Наступним критичним етапом є порівняльний і крос-дисциплінарний аналіз змісту. Дослідник вручну вивчає кожну публікацію, виокремлюючи постановку проблеми, методологію, отримані результати та висновки. Для структурування цієї інформації часто створюються спеціальні таблиці або матриці, що дозволяють порівнювати підходи різних авторів, виявляти суперечності та консенсус. Паралельно ведеться детальне ведення нотаток та написання анотованих рефератів, що слугують особистим знаннєвим архівом дослідника і основою для майбутнього тексту огляду літератури. Фінальним етапом є тематична класифікація та синтез, коли джерела групуються за концептуальними напрямами, методами або результатами. Це дозволяє виявити основні дослідницькі кластери, тенденції та, що найважливіше, лакуни в існуючих знаннях. Весь цей комплекс ручних операцій вимагає високої концентрації, аналітичних здібностей та часу, перетворюючись на мистецтво інтерпретації та структурування знань.

1.2.3 Обмеження традиційних підходів

Незважаючи на свою фундаментальну роль, традиційні методи наукового пошуку та аналізу літератури демонструють вагомі обмеження в умовах сучасних викликів. Найбільш очевидною є проблема масштабованості та ефективності. Швидкість генерування нових знань у телекомунікаціях робить повний ручний огляд великих масивів публікацій (наприклад, для систематичного огляду) практично неможливим за розумні часові рамки. Це призводить до того, що дослідники часто змушені обмежуватися вибірковим аналізом, що підвищує ризик пропуску важливих джерел. Цю проблему посилює неповнота охоплення навіть найбільших баз даних. Матеріали локальних конференцій, технічні звіти промислових компаній, дисертації та публікації українською мовою нерідко залишаються за межами глобальних індексів, створюючи «сліпі плями» та упередження в уявленні про стан досліджень у певному регіоні чи ніші.

Суб’єктивність та упередження є іншим серйозним обмеженням. Відбір

джерел, формулювання пошукових запитів і інтерпретація результатів неминуче знаходяться під впливом освіти, досвіду та наукової школи дослідника. Це може призводити до підтверджувального упередження, коли непомітно надається перевага роботам, що підтверджують власні гіпотези, а альтернативні точки зору ігноруються. Особливо це проявляється в міждисциплінарних дослідженнях, де необхідність шукати літературу в суміжних галузях (наприклад, економіці або соціології для дослідження прийняття 5G) вимагає володіння сторонньою термінологією та методологією, що є значною перешкодою для дослідника-«технаря».

Критичним для сучасної науки є також питання відтворюваності та прозорости процесу пошуку. Деталі стратегії пошуку, критерії включення та виключення джерел рідко документуються в повному обсязі та публікуються разом із науковою роботою. Це робить літературний огляд майже неможливим для точного відтворення іншим дослідником, знижуючи довіру до отриманих результатів. Нарешті, інтенсивний ручний аналіз на тлі інформаційної надмірності призводить до значного когнітивного навантаження. Необхідність одночасно утримувати в пам’яті велику кількість концепцій, деталей та контекстів з різних джерел підвищує ймовірність помилок, втрати важливих зв’язків та загального виснаження дослідника, що впливає на якість підсумкового синтезу знань. Таким чином, традиційна методологія, залишаючись цінною основою, все більше потребує доповнення новими інструментами та підходами для подолання цих системних обмежень.

1.3 Сучасні технології ШІ в науковій діяльності

Стрімке зростання обсягів даних, складність міждисциплінарних завдань і висока конкурентність підвищують вимоги до швидкості та якості наукового пошуку, аналізу та синтезу. Сучасні ШІ-технології, зокрема великі мовні моделі, спеціалізовані наукові асистенти та автономні дослідницькі агенти, пропонують нові парадигми роботи, спрямовані на подолання обмежень традиційних методів. Вони дозволяють автоматизувати рутинні, трудомісткі операції, виявляти складні закономірності в великих масивах літератури та даних, а також забезпечувати

підтримку на всіх етапах наукової роботи — від формування гіпотези до підготовки публікації. Цей розділ присвячений аналізу основних напрямів застосування цих технологій, оцінки їх потенціалу та викликів, які супроводжують їх впровадження в академічну практику.

1.3.1 Системи на основі LLM загального призначення

Великі мовні моделі, такі як ChatGPT, Claude та Gemini, репрезентують собою радикально новий інструментарій для науковця, що ґрунтується на глибокому навчанні та обробці природної мови. Їхня фундаментальна відмінність від традиційних пошукових систем полягає в здатності не просто знаходити документи за ключовими словами, а розуміти семантику запиту, синтезувати інформацію з різних джерел та генерувати зв’язний, контекстуально релевантний текст [2, 8]. У телекомунікаціях це відкриває широкі можливості для прискорення та покращення якості наукової роботи. Моделі можуть виконувати швидкий попередній аналіз великих масивів наукових статей, формуючи стислі огляди за заданою темою, наприклад, щодо архітектури мереж 6G, методів захисту квантового ключового розподілу або трендів у розвитку SDN. Це дозволяє дослідникові за лічені години отримати структуроване уявлення про стан проблеми, ідентифікувати ключові роботи та визначити прогалини у дослідженнях.

Окрім пасивного аналізу, LLM активно залучаються до творчих та технічних аспектів дослідження. Вони здатні допомагати у формулюванні наукових гіпотез на основі виявлених закономірностей у літературі, пропонувати експериментальні методики або планувати етапи дослідження. Для інженерної частини роботи їхня здатність генерувати, коментувати та налагоджувати програмний код на Python [7], MATLAB, R чи інших мовах є незамінною при створенні симуляційних моделей мереж, обробці сигналів, візуалізації даних та проведенні статистичного аналізу. Також LLM слугують потужними інструментами для наукової комунікації: вони можуть допомагати у написанні та редагуванні статей, технічних звітів, грантових заявок, автоматично покращуючи стилістику, узгоджуючи термінологію та адаптуючи текст під вимоги конкретного видання. Інтеграція цих моделей у платформи спільної роботи (наприклад, через API в Jupyter Notebooks або текстові

редактори) дедалі більше впроваджує їх у повсякденний робочий процес дослідницьких колективів [2, 9, 10].

Однак ефективне використання LLM супроводжується низкою суттєвих застережень та обмежень. Найкритичнішим є проблема «галюцинацій» або генерації правдоподібної, але фактично невірної або вигаданої інформації, особливо коли мова йде про вузькоспеціалізовані технічні деталі або найновіші, ще недостатньо відображені в тренувальних даних, розробки. Моделі можуть неправильно інтерпретувати контекст, давати застарілі відомості щодо стандартів чи протоколів або пропонувати неможливі з інженерної точки зору рішення. Крім того, їхній аналіз часто є поверхневим і не може замінити глибокого експертного розуміння фізичних принципів або математичної строгості. Питання конфіденційності даних також залишається актуальним, особливо при роботі з непублічною інформацією про мережеві конфігурації або комерційні алгоритми. Таким чином, LLM слід розглядати не як автономних експертів, а як потужних асистентів, чия робота вимагає обов’язкового критичного контролю, верифікації та доопрацювання з боку кваліфікованого дослідника.

1.3.2 Спеціалізовані наукові асистенти та інструменти

На противагу універсальним LLM, на ринку з’являються спеціалізовані ШІ-інструменти, сконцентровані на вирішенні конкретних наукових завдань, що робить їх значно ефективнішими у своїй ніші. Серед них вирізняються такі платформи, як Elicit, Scite та Consensus, кожна з яких трансформує певний етап роботи з літературою. Elicit позиціонується як «дослідницький помічник» для проведення систематичних оглядів. Замість того щоб просто повертати список статей за запитом, система використовує ШІ для автоматичного вилучення ключової інформації з сотень публікацій: мети дослідження, методології, розміри вибірки, основні результати та висновки. На основі цього Elicit може генерувати зведені таблиці порівняння, що дозволяє дослідникові швидко провести мета-аналіз, виявити закономірності в методах або суперечності в результатах без необхідності вручну конспектувати кожну роботу. Це кардинально підвищує продуктивність на початковому етапі роботи та зменшує ризик пропустити важливі

деталі через людську втому.

Інструмент Scite вносить революційні зміни в практику аналізу цитувань. Традиційні метрики (на кшталт h-індексу чи кількості цитувань) відображають лише популярність, але не зміст посилань. Scite за допомогою ШІ аналізує текст цитування, класифікуючи його на такі категорії, як «підтримуюче», «констатуюче» або «спростовуюче». Це дозволяє дослідникові оцінити реальний науковий вплив та суперечливість публікації, виявити дискусійні моменти в науці та відстежити, як певний результат підтверджується або критикується в подальших дослідженнях. Для швидко розвиваючоїся галузі телекомунікацій, де нові технології часто супроводжуються гострими науковими дискусіями, такий контекстуальний аналіз є надзвичайно цінним для оцінки надійності джерел.

Consensus працює як «пошукова система з доказовими відповідями». Користувач ставить прямий наукове запитання (наприклад, «Який протокол маршрутизації є найбільш енергоефективним для сенсорних мереж IoT?»), і система, замість посилань на статті, надає стислу витягнуту з рецензованих джерел відповідь, супроводжуючи її прямими цитатами. Це дозволяє швидко отримати науково обґрунтовану довідку, мінімізуючи час на самостійний пошук та синтез. Подібні інструменти особливо корисні для міждисциплінарних досліджень, коли дослідник працює в новій для себе субгалузі та потребує швидкого занурення в проблематику. Незважаючи на їхню ефективність, важливо розуміти, що якість роботи таких асистентів залежить від якості та актуальності їхньої власної бази знань і алгоритмів обробки, тому вони також потребують критичного осмислення отриманих результатів.

1.3.3 Підходи до автоматизації наукового пошуку та "research agents"

Наступним логічним кроком у еволюції ШІ для науки є розвиток автономних або напівавтономних дослідницьких агентів (research agents). Це вже не просто інструменти пошуку чи аналізу, а складні системи, здатні виконувати багатоетапні дослідницькі завдання з мінімальним втручанням людини. Вони базуються на комбінації передових технологій: спеціалізованих мовних моделей (наприклад, SciBERT, навчених на наукових текстах), механізмів посилення генерації за

допомогою вибірки (Retrieval-Augmented Generation, RAG), графових баз знань для аналізу зв’язків між публікаціями та методів підкріплювального навчання для прийняття рішень [1, 12].

У контексті телекомунікацій research agents можуть бути запрограмовані на виконання таких комплексних завдань. Наприклад, агент може отримати завдання «проаналізувати тенденції розвитку алгоритмів машинного навчання для оптимізації трафіку в 5G-мережах за останні 5 років». Він самостійно формує та виконує низку дій: проводить пошук у відповідних базах даних (IEEE Xplore, arXiv), фільтрує та класифікує знайдені роботи, витягує ключові параметри алгоритмів та їхню ефективність, будує граф цитувань для виявлення ключових впливових статей, синтезує зведений звіт із візуалізаціями та навіть може запропонувати перспективні напрямки для подальших досліджень на основі виявлених прогалин. Більш просунуті агенти можуть переходити від теоретичного аналізу до експериментального планування: на основі літератури пропонувати параметри симуляцій, генерувати відповідний код, запускати моделювання та інтерпретувати його результати.

Особливий потенціал мають багатоагентні системи, де різні спеціалізовані агенти (пошуковик, аналітик даних, експерт з моделювання, редактор звіту) співпрацюють між собою, розподіляючи складні завдання [5, 18]. Такі системи можуть значно підвищити відтворюваність досліджень, оскільки кожен крок може бути детально залогований агентом. Однак розвиток цього напрямку стикається з серйозними викликами, включаючи потребу у великих обчислювальних ресурсах, складність забезпечення прозорості та інтерпретованості рішень, прийнятих агентом, а також ризик посилення наявних упереджень у тренувальних даних [6, 15, 17]. Крім того, повна автономія агента в генерації гіпотез та плануванні експериментів поки що є предметом фундаментальних досліджень і потребує ретельного контролю з боку людини-експерта. Незважаючи на це, research agents представляють собою майбутнє автоматизації науки, обіцяючи не лише прискорити рутинні процеси, але й допомогти у відкритті нових, неочевидних зв’язків у великомасштабних наукових знаннях.

1.4 Сфери застосування технологій ШІ в наукових дослідженнях

Інтеграція штучного інтелекту в наукові дослідження не обмежується лише пошуком та аналізом літератури. Вона охоплює весь дослідницький цикл — від планування експерименту до публікації результатів, трансформуючи методологію роботи в фундаментальних та прикладних науках. Умови сучасної наукової діяльності, такі як експоненційне зростання даних, міждисциплінарність і високі вимоги до швидкості отримання результатів, роблять автоматизацію не бажаною, а необхідною. ШІ виступає як універсальний інструмент, здатний адаптуватися до специфіки різних етапів дослідження, пропонуючи рішення для систематизації знань, обробки складних датасетів, оптимізації експериментів та академічного письма. Цей розділ детально розглядає ключові сфери застосування ШІ, зокрема автоматизацію літературних оглядів, аналіз та візуалізацію даних, генерацію наукових текстів, а також специфіку їх імплементації в контексті телекомунікаційних досліджень, де технологічні виклики набувають особливої гостроти.

1.4.1 Автоматизація систематичного огляду літератури

Систематичний огляд літератури (SLR) є одним з найбільш трудомістких та методологічно вимогливих етапів наукового дослідження. Автоматизація цього процесу за допомогою ШІ значно підвищує його ефективність, об'єктивність і відтворюваність. Основу цієї автоматизації складають передові технології обробки природної мови (NLP), такі як трансформерні архітектури (BERT, SciBERT, RoBERTa), спеціально донавчені на великих корпусах наукових текстів. Ці моделі здатні розуміти семантичні зв'язки між технічними термінами, що дозволяє не просто шукати за ключовими словами, а виявляти концептуально релевантні публікації навіть при відмінності в термінології.

Процес автоматизованого SLR зазвичай включає кілька етапів, підтримуваних спеціалізованими платформами. На етапі первинного скринінгу інструменти на кшталт ASReview або Rayyan використовують алгоритми активного навчання. Система пропонує дослідникові для огляду ті статті, щодо релевантності яких її модель найменш впевнена, що дозволяє швидко навчитися

критеріям відбору та значно скоротити загальний час скринінгу тисяч заголовків і анотацій. Для поглибленого аналізу та вилучення даних астосовуються системи типу Elicit або IRIS.ai. Вони автоматично аналізують повні тексти статей, вилучаючи структуровану інформацію: мету дослідження, методологію, розмір вибірки, ключові результати, обмеження. Ці дані потім автоматично агрегуються в порівняльні таблиці або матриці, що є основою для мета-аналізу.

Окремо варто відзначити роль LLM, таких як GPT-4, у якості асистентів для синтезу. Вони можуть аналізувати вилучені дані, формулювати стислі резюме по кожній групі досліджень, виявляти суперечності між результатами різних авторів та навіть пропонувати висновки щодо загальних тенденцій. Проте, основним викликом залишається проблема доменної специфічності та "чорної скриньки". Моделі, загально навчені, можуть недостатньо добре розуміти тонкощі вузькогалузевих термінів телекомунікацій (наприклад, деталі нового протоколу зв'язку чи архітектури O-RAN), що може призводити до пропуску важливих робіт або неправильної інтерпретації методів. Тому автоматизований огляд залишається напівавтоматичним процесом, де заключний етап критичної оцінки, інтеграції знань та формулювання власних висновків беззаперечно належить досліднику-людині.

1.4.2 Аналіз та візуалізація складних даних

Сучасні телекомунікаційні дослідження генерують масивні, швидкозмінні та структурно складні набори даних: від часових рядів мережевого трафіку і сигналів фізичного рівня до лог-файлів мільйонів пристроїв IoT та соціотехнічних метрик користувацької поведінки. Традиційні методи статистичного аналізу часто недостатні для виявлення нетривіальних залежностей і патернів у таких даних. ШІ, зокрема методи глибокого навчання, пропонує потужний інструментарій для їхнього аналізу та інтуїтивної візуалізації.

Для аналізу активно використовуються різноманітні архітектури нейронних мереж. Згорткові нейронні мережі (CNN) ефективні для обробки просторових даних, наприклад, спектрограм сигналів або геопросторових карт покриття мережі. Рекурентні нейронні мережі (RNN), особливо їхні модифікації з довгостроковою

короткочасною пам’яттю (LSTM), незамінні для аналізу послідовних даних, таких як прогнозування завантаження мережі в часі або виявлення аномалій у трафіку. Трансформери, завдяки механізму уваги, досягають високої ефективності в обробці довгих послідовностей і мультимодальних даних (наприклад, поєднання логів, метрик продуктивності та текстових звітів).

Інструменти для візуалізації, такі як KNIME, Orange або Tableau, інтегрують ШІ-моделі для автоматичного створення інтерактивних дашбордів, графів зв'язків (network graphs) та багатовимірних проекцій. Вони дозволяють не лише відобразити результати аналізу, але й здійснювати "внутрішнє дослідження" даних: користувач може взаємодіяти з візуалізацією, виділяти кластери, фільтрувати параметри, і система в реальному часі перераховує моделі та узагальнення. У телекомунікаціях це застосовується для візуалізації топології та стану мереж, діагностики аномалій безпеки у вигляді теплових карт або діаграм потоків, а також для представлення результатів оптимізації розміщення базових станцій [6].

Головними викликами в цій сфері є масштабованість обчислень, пояснюваність та якість даних. Моделі, натреновані на синтетичних або історичних даних, можуть погано працювати в реальному часі в нових умовах мережі, що вимагає постійного донавчання. Крім того, для прийняття обґрунтованих рішень (наприклад, у разі виявлення атаки) інженери повинні розуміти, чому модель видала певний результат, що є складною задачею для складних архітектур глибокого навчання.

1.4.3 Написання та форматування наукових текстів

ІПідготовка наукових публікацій — статей, звітів, дисертацій — є значною частиною академічного навантаження. ШІ-інструменти для написання та форматування текстів спрямовані на автоматизацію рутинних аспектів цієї роботи, звільняючи час дослідника для творчого аналізу та інтерпретації результатів. Великі мовні моделі стали ядром цих інструментів. Вони можуть допомагати на різних стадіях письма: від генерації початкового нарисуокремих розділів (наприклад, "Методологія" або "Обговорення") на основі наданих даних і конспектів, до поліпшення стилю, перефразування незрозумілих речень та

автоматичної корекції граматики.

Спеціалізовані платформи, такі як PaperPal, Writefull або SciSpace, пропонують функції, адаптовані під академічний контекст. Вони можуть перевіряти текст на відповідність академічному стилю, пропонувати більш точні наукові терміни, виявляти можливі плагіат та перевіряти коректність цитувань. Для форматування інструменти інтегровані з LaTeX-середовищами на кшталт Overleaf, пропонуючи шаблони відповідно до вимог конкретних видавництв (IEEE, ACM, Springer). Користувач може зосередитись на змісті, а система автоматично сформує бібліографію, підписи до рисунків та таблиць, і навіть заповнить метадані статті.

У телекомунікаціях, де публікації часто містять складні технічні описи, математичні формули, алгоритми та специфікації, ШІ може допомогти у генерації пояснювального тексту для кодів симуляцій, створенні технічних звітів на основі лог-файлів експериментів або навіть у написанні патентних заявок. Однак ключовим обмеженням залишається наукова валідація. Модель може створити граматично ідеальний, але технічно помилковий або спрощений опис. Генерація формул або технічних деталей без контролю може призвести до серйозних неточностей. Тому роль дослідника як експерта, який перевіряє, редагує та затверджує кожен згенерований фрагмент, залишається критичною. Ризики неправомірного запозичення ідей (плагіату) або втрати авторського голосу також вимагають уважного ставлення до використання таких інструментів.

1.4.4 Специфіка застосування в галузі телекомунікацій

Застосування ШІ в телекомунікаційних дослідженнях має низку унікальних особливостей, обумовлених технологічною специфікою самої галузі. По-перше, це робота з даними надзвичайно великого обсягу та швидкості (Big Data та Fast Data). Мережі 5G/6G та IoT генерують петабайти інформації в реальному часі, що вимагає від ШІ-рішень не лише масштабованості, але й здатності працювати на периферії мережі (Edge AI) для швидкого прийняття рішень з низькою затримкою. По-друге, висока міра конфіденційності та безпеки даних обмежує доступ до реальних операторських даних для навчання моделей, спонукаючи до розвитку методів федеративного навчання та генерації реалістичних синтетичних даних [3, 4].

Ключовою тенденцією є пряма інтеграція ШІ в телекомунікаційну інфраструктуру, а не лише її використання як зовнішнього аналітичного інструменту. Яскравим прикладом є архітектура O-RAN, зокрема компонент RAN Intelligent Controller (RIC). RIC надає платформу, на якій сторонні розробники можуть розгортати спеціалізовані ШІ-додатки (xApps, rApps) для керування мережевими функціями в реальному часі — динамічного розподілу ресурсів, керування енергоспоживанням, виявлення аномалій. Це перетворює мережу зі статичної на самоорганізовану та відкриває нові горизонти для наукових експериментів у "живих" умовах.

Окрім O-RAN, ШІ активно використовується в дослідженнях з оптимізації протоколів зв'язку, де генетичні алгоритми або навчання з підкріпленням допомагають знаходити оптимальні параметри конфігурації. У бездротових комунікаціях ШІ застосовується для компресії каналу, адаптивної модуляції та кодування, а також для керування масивними MIMO-антенами. Проте, специфічними викликами для галузі залишаються проблема пояснюваності рішень ШІ в критично важливих мережах, забезпечення стійкості до ворожих атак на самі ШІ-моделі, а також необхідність враховувати фізичні обмеження апаратного забезпечення при розробці алгоритмів. Таким чином, успішна інтеграція ШІ в телекомунікаційні дослідження вимагає глибокого симбіозу між експертами з телекомунікацій, дата-сайентистами та фахівцями з кібербезпеки [15].

Висновки

Сучасні дослідження в телекомунікаціях стикаються із серйозними системними викликами, такими як експоненційне зростання інформації, необхідність міждисциплінарного аналізу та підвищені вимоги до академічної строгості та відтворюваності результатів. Традиційні методи пошуку та аналізу літератури в цих умовах демонструють значні обмеження: низьку масштабованість, суб'єктивність, неповне охоплення джерел та велике когнітивне навантаження на дослідника. Це створює нагальну потребу у нових підходах до автоматизації наукової роботи.

Технології штучного інтелекту, зокрема великі мовні моделі та спеціалізовані

наукові асистенти, пропонують трансформаційні рішення для автоматизації рутинних операцій, аналізу складних даних і підтримки у написанні текстів. Однак їхнє застосування супроводжується ризиками, як-от генерація неточної інформації та залежність від якості даних, що вимагає обов'язкового критичного контролю з боку дослідника. Таким чином, ШІ-інструменти слід розглядати не як автономних експертів, а як потужних асистентів.

У контексті телекомунікацій застосування ШІ набуває специфіки через роботу з даними надзвичайного обсягу та швидкості, високі вимоги безпеки та пряму інтеграцію в мережеву інфраструктуру (наприклад, O-RAN). Незважаючи на перспективи автоматизації всього дослідницького циклу, ключовими викликами залишаються пояснюваність рішень ШІ, забезпечення стійкості до атак та необхідність тісного симбіозу між фахівцями з телекомунікацій, дата-саєнсу та кібербезпеки для успішної реалізації цього потенціалу.

РОЗДІЛ 2

АРХІТЕКТУРИ ТА ТЕХНОЛОГІЇ ПОБУДОВИ СПЕЦІАЛІЗОВАНИХ ШІ-АГЕНТІВ ДЛЯ НАУКОВОГО АНАЛІЗУ

2.1 Архітектурні підходи до створення автономних ШІ-агентів

2.1.1 Модель "Мислення-Дії-Спостереження"

У сучасних дослідженнях зі штучного інтелекту, особливо в контексті наукового аналізу у телекомунікаційній сфері, ключовою архітектурною парадигмою стала модель. Ця модель забезпечує концептуальну основу для побудови автономних агентів, здатних до ітеративного та адаптивного вирішення складних завдань. В її основі лежить циклічний процес, що імітує класичний науковий метод: формулювання гіпотези (мислення), її експериментальну перевірку (дія) та аналіз отриманих результатів (спостереження). Такий підхід дозволяє агенту не лише реагувати на зміни в середовищі, але й самостійно вдосконалювати свої стратегії на основі накопиченого досвіду [5].

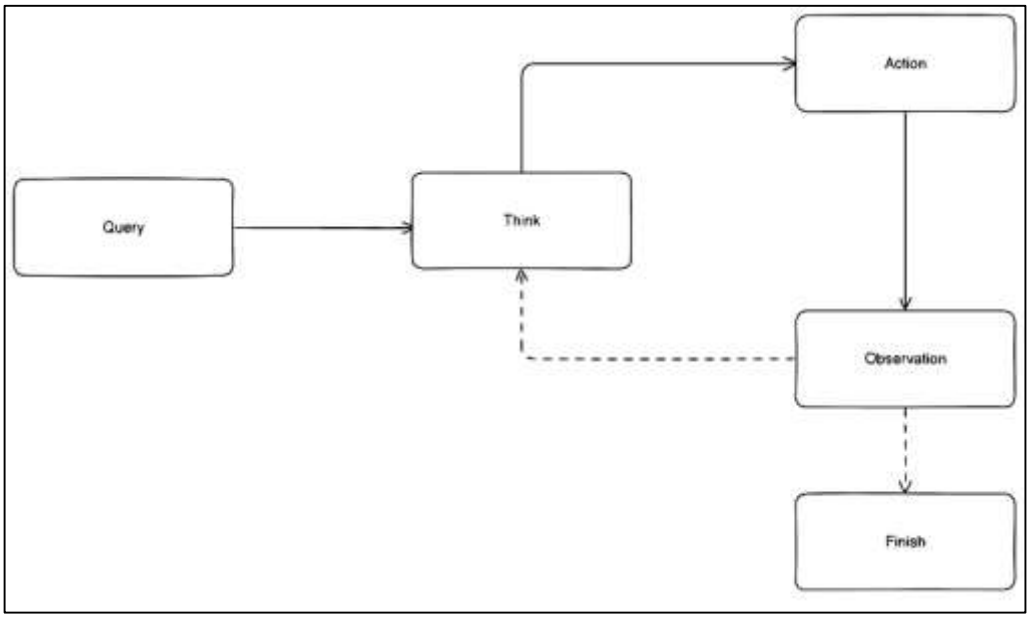

Рис.2.1. Блок-схема моделі RAO

Етап мислення (Reasoning) передбачає глибокий аналіз поточного стану системи, наявних даних та контексту завдання. У телекомунікаціях це може включати оцінку топології мережі, динаміки трафіку, показників якості послуг або виявлення потенційних аномалій. На цьому етапі агент формує гіпотези, будує

плани дій або прогнози, використовуючи як внутрішні знання, так і зовнішні моделі та бази даних. Важливою характеристикою є здатність до абстрактного мислення та планування послідовностей кроків для досягнення поставленої дослідницької цілі.

Наступний етап — дія (Action) — є практичною реалізацією сформованого плану. Агент взаємодіє з дослідницьким середовищем через спеціалізовані інструменти. У телекомунікаційному контексті це може бути налаштування параметрів обладнання, ініціювання вимірювань, запуск симуляційних моделей або активний збір даних з різних джерел. Ефективність цього етапу безпосередньо залежить від інтеграції агента з відповідними програмними та апаратними інтерфейсами, що забезпечують виконання конкретних операцій.

Завершальною ланкою циклу є спостереження (Observation). Після виконання дії агент отримує нові дані, що відображають результат його втручання. Ця інформація ретельно інтерпретується: оцінюються наслідки, порівнюються з очікуваннями, виявляються закономірності або аномалії. Результати спостережень формалізуються та зберігаються в структурі пам'яті агента (наприклад, у вигляді логів, оновлених індексів або баз знань), формуючи досвід для наступних ітерацій. Цей крок замикає цикл, забезпечуючи основу для корекції гіпотез і планів, що і є суттю адаптивного навчання.

Переваги моделі RAO у науковому аналізі, зокрема в телекомунікаціях, є значними. Вона забезпечує високу ступінь формалізації дослідницького процесу, що спрощує реплікацію експериментів і верифікацію результатів. Модель природно підтримує модульність, дозволяючи чітко розділяти логіку мислення, механізми виконання дій та системи зберігання досвіду. Крім того, вона є ідеальною основою для побудови мультиагентних систем, де кілька автономних одиниць можуть співпрацювати, вирішуючи складні, розподілені завдання, такі як оптимізація мережі зв'язку чи масштабний моніторинг якості послуг. Однак практична реалізація RAO-агентів вимагає вирішення низки інженерних викликів, серед яких розробка потужних та ефективних модулів мислення, організація надійної та масштабованої пам'яті, а також забезпечення ефективного управління

життєвим циклом агента в умовах обробки великих обсягів даних і розподілених обчислень.

2.1.2 Фреймворки для побудови агентів

Для практичної реалізації автономних ШІ-агентів, побудованих на принципах RAO, існує низка спеціалізованих програмних фреймворків. Найбільш поширеними та адаптованими для наукового аналізу, зокрема в телекомунікаційній галузі, є LangChain, LlamaIndex та Haystack. Кожен з них пропонує власний архітектурний погляд на оркестрацію агентів, управління даними та реалізацію ітеративного циклу, залишаючись при цьому гнучким інструментом для дослідників.

LangChain орієнтований на концепцію ланцюгів (chains) та агентів як центральних оркестраторів. Його архітектура дозволяє створювати складні конвеєри обробки, де агенти приймають рішення про послідовність викликів інструментів на основі динамічного контексту.

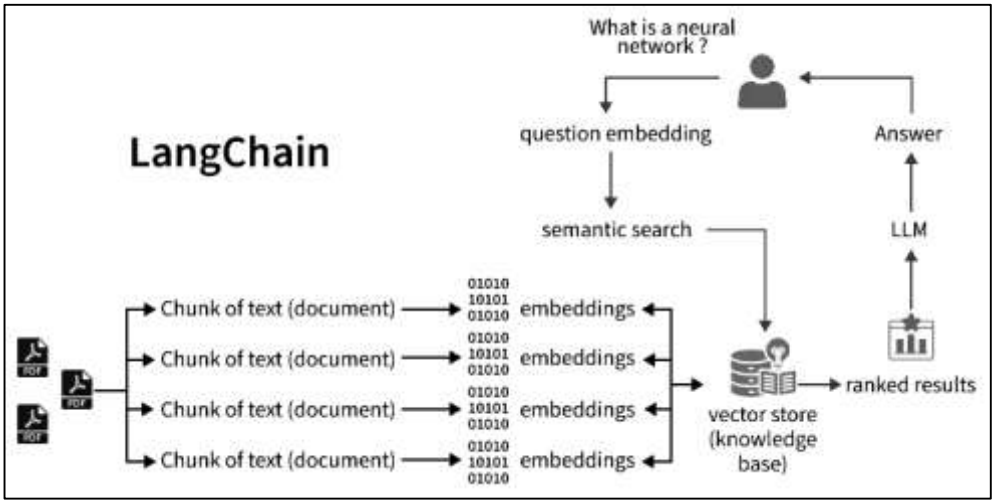

Рис.2.2. Загальний алгоритм логіки LangChain

Ключовою перевагою LangChain є його модульність: компоненти, такі як агенти, ланцюги, набори інструментів та системи пам'яті, є ізольованими та взаємозамінними. Це спрощує тестування, налагодження та адаптацію системи до нових дослідницьких сценаріїв. Для реалізації етапу мислення LangChain активно використовує шаблони, такі як ReAct (Reasoning + Acting), що поєднує логічні міркування з пошуковими діями, або Plan-and-Execute, де спочатку будується

детальний план, а потім він виконується. Система пам'яті (наприклад, ConversationBufferMemory або SummaryMemory) дозволяє агенту зберігати контекст протягом багатоетапних сесій, що критично важливо для довготривалих експериментів у телекомунікаціях, таких як тривалий моніторинг або поетапна оптимізація параметрів мережі.

LlamaIndex, на відміну від LangChain, зосереджений навколо ефективної індексації та структурованого представлення даних для LLM. Його архітектура ідеально підходить для сценаріїв, де робота агента тісно пов'язана з пошуком, аналізом і синтезом інформації з великих корпусів наукових даних, документації або результатів попередніх експериментів.

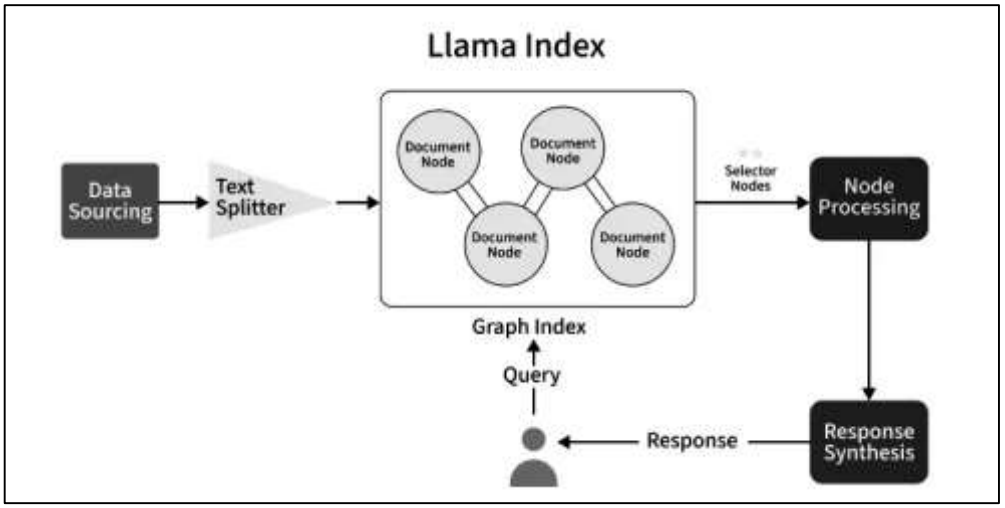

Рис.2.3. Загальний алгоритм логіки LlamaIndex

Основним компонентом є індекс, який організовує дані у ієрархічній структурі. Агент у LlamaIndex взаємодіє з цими даними через механізми запитів (Query Engines) та пошуку (Retrievers), що становить основу його етапу мислення. На основі отриманої інформації агент формує план дій. Таким чином, LlamaIndex природним чином інтегрує фазу спостереження у вигляді постійного оновлення та звернення до індексів, що робить його особливо корисним для досліджень, що базуються на літературі, історичних даних або структурованих технічних звітах у телекомунікаціях.

Haystack пропонує третій, пайплайновий підхід до архітектури агентів. Його основа — це направлені ациклічні графи, де кожен вузол представляє окремий етап

обробки: пошук, читання, генерація тексту, прийняття рішень тощо. Така структура дозволяє явно моделювати складні, розгалужені дослідницькі протоколи з паралельним виконанням завдань та різними шляхами обробки даних. Агент у Haystack може бути реалізований як окремий пайплайн або як контролер, що керує потоком між вузлами. Це надає високу гнучкість для організації як простих, так і ієрархічних сценаріїв, наприклад, коли необхідно паралельно збирати дані з різних сегментів телекомунікаційної мережі, а потім інтегрувати їх для комплексного аналізу. Управління контекстом і пам'яттю відбувається за рахунок явної передачі даних між вузлами, що забезпечує прозорість і відтворюваність кожного кроку в циклі RAO.

Для наочності ключові архітектурні відмінності та характеристики цих фреймворків представлені в таблиці 2.1.

Таблиця 2.1

Порівняльна характеристика фреймворків для побудови ШІ-агентів

Аспект	**LangChain**	**LlamaIndex**	**Haystack**
Основна парадигма	Ланцюги (Chains) та агенти-оркестратори	Індексно-орієнтована архітектура для роботи з даними	Пайплайни на основі направлених ациклічних графів (DAG)
Реалізація Reasoning	Шаблони (ReAct, Plan-and-Execute), динамічне планування на основі контексту	Пошук та аналіз у структурованих індексах, формування відповідей на запити	Логіка, розподілена між спеціалізованими вузлами (нодами) пайплайну
Реалізація Action	Виклики модульних інструментів (Tools) через агенти	Виконання дій через інтеграцію з інструментами на основі результатів пошуку	Виконання операцій у вигляді задач конкретних вузлів пайплайну
Пам'ять	Вбудовані системи (Buffer, Summary), прив'язані до агента	Контекст на основі сесії, історія взаємодії з індексами	Явна передача контексту та стану між вузлами пайплайну

Продовження таблиці 2.1

Аспект	LangChain	LlamaIndex	Haystack
Модульність	Дуже висока, компоненти легко замінюються та комбінуються	Висока, принцип plug-and-play для індексів, ретріверів, обробників	Висока, кожен вузол пайплайну є незалежним модулем
Сильні сторони	Гнучкість створення складних логік взаємодії, підтримка багатьох LLM	Ефективна робота з великими обсягами структурованих даних, швидкий пошук	Масштабованість, підтримка складних розгалужених протоколів обробки
Оптимальні сценарії	Динамічні діалогові агенти, системи з комплексним плануванням дій	Дослідження, що базуються на аналізі документів, технічних звітів, баз знань	Великі наукові експерименти з чітко визначеними етапами, паралельна обробка

Вибір конкретного фреймворку залежить від специфіки дослідницького завдання в телекомунікаційній сфері. LangChain є універсальним інструментом для створення інтерактивних агентів зі складною логікою прийняття рішень. LlamaIndex незамінний там, де ядром роботи є ефективне управління та аналіз великих масивів структурованої інформації. Haystack, своєю чергою, є оптимальним рішенням для впорядкованих, багатоетапних експериментальних протоколів, де важлива чіткість, масштабованість та можливість паралелізації задач. Усі вони, використовуючись окремо або в комбінації, дозволяють реалізувати потужні автономні системи для автоматизації наукового аналізу, здатні проводити експерименти, обробляти дані та генерувати знання в складному динамічному середовищі телекомунікацій.

2.2 Технології семантичного пошуку та роботи з науковими документами

2.2.1 Векторні бази даних та пошук за подібністю

Основним викликом сучасних наукових досліджень у телекомунікаціях є ефективна навігація та аналіз величезних масивів структурованої та неструктурованої інформації, такої як наукові статті, патенти, технічні стандарти та специфікації. Традиційні системи пошуку за ключовими словами часто виявляються недостатніми через складність галузевої термінології, синонімію та контекстну залежність понять. Відповіддю на ці виклики стали технології семантичного пошуку, фундаментом яких є векторні бази даних [12]. Векторні бази даних зберігають інформацію у вигляді числових векторів — їхніх семантичних уявлень (ембеддингів), отриманих за допомогою передових нейромережевих моделей, таких як BERT, SciBERT чи спеціалізованих трансформерів для наукових текстів. Це дозволяє здійснювати пошук не за формальним збігом слів, а за змістовною подібністю, знаходячи документи, що обговорюють схожі концепції навіть при використанні різної термінології. Такі системи критично важливі для телекомунікацій, де швидка еволюція технологій вимагає оперативного виявлення зв'язків між новими протоколами, патентами та академічними дослідженнями.

Серед провідних відкритих рішень для векторного пошуку виділяються ChromaDB, Weaviate та Qdrant, кожне з яких пропонує унікальні архітектурні переваги [12, 14]. ChromaDB відрізняється простотою інтеграції та гнучкістю, будучи орієнтованою на швидке прототипування та дослідницькі проекти. Вона ефективно працює з алгоритмом HNSW для швидкого пошуку найближчих сусідів і підтримує розширену фільтрацію за метаданими, такими як автор, дата публікації або тип документа, що є ключовим для структурованого наукового корпусу. Weaviate, з іншого боку, виходить за рамки простої векторної бази, інтегруючи елементи графової бази знань. Її мікросервісна архітектура та підтримка GraphQL дозволяють моделювати складні взаємозв'язки між сутностями (наприклад, між авторами, установами, поняттями та цитатами), що робить її ідеальною для побудови комплексних наукових онтологій у телекомунікаціях. Qdrant

відзначається високою продуктивністю та орієнтацією на розподілені, навантажені середовища завдяки реалізації на мові Rust. Вона забезпечує масштабованість, надійність та низькі затримки при роботі з сотнями мільйонів векторів, що критично для великих міжнародних репозиторіїв наукової літератури.

Порівняльні характеристики цих систем наведені в таблиці 2.2.

Таблиця 2.2

Порівняльна характеристика векторних баз даних даних

Критерій	ChromaDB	Weaviate	Qdrant
Основна перевага	Простота, швидка інтеграція, ідеальна для прототипів	Графова модель знань, гнучка схема, потужні API	Висока продуктивність, масштабованість, Rust-ядро
Алгоритм пошуку	HNSW	HNSW та власні індекси	HNSW з оптимізаціями для швидкодії
Метадані та фільтри	Підтримка фільтрації за метаданими	Розширена схема властивостей, GraphQL-фільтри	Гнучкі фільтри, підтримка складних умов
Масштабованість	Базова шардінг-підтримка	Горизонтальне масштабування, кластеризація	Горизонтальне масштабування, реплікація, шардінг
Інтеграція з ШІ-екосистемою	Широка через Python-бібліотеки	Вбудована підтримка моделей, модульність	API-first підхід, сумісність з різними пайплайнами

Ефективне застосування цих технологій у телекомунікаційних дослідженнях вимагає не лише вибору інструменту, але й створення якісних векторних уявлень. Для цього використовуються спеціалізовані моделі ембеддингів, натреновані на наукових текстах (наприклад, SciBERT), що краще розуміють технічну термінологію, математичні формули та формальні описи протоколів. Це забезпечує високу точність пошуку при роботі з такими документами, як специфікації 3GPP, стандарти IEEE або патенти у галузі бездротового зв'язку.

2.2.2 Техніки RAG для забезпечення релевантності відповідей

Незважаючи на потужність сучасних LLM, їхня основна слабкість у науковому контексті полягає в обмеженості знань фіксованим навчальним корпусом, що може швидко застарівати у динамічній галузі телекомунікацій. Техніка RAG елегантно вирішує цю проблему, поєднуючи досягнення генеративних моделей з динамічним доступом до зовнішніх баз знань [1]. Архітектурно RAG складається з двох ключових компонентів: ретрівера (retriever) та генератора (generator). Ретривер, зазвичай побудований на основі векторної бази даних, здійснює семантичний пошук найбільш релевантних фрагментів документів у відповідь на запит користувача. Ці фрагменти, разом із оригінальним запитом, формують збагачений контекст, який потім подається на вхід генератору — великій мовній моделі, яка синтезує когерентну, інформативну та аргументовану відповідь.

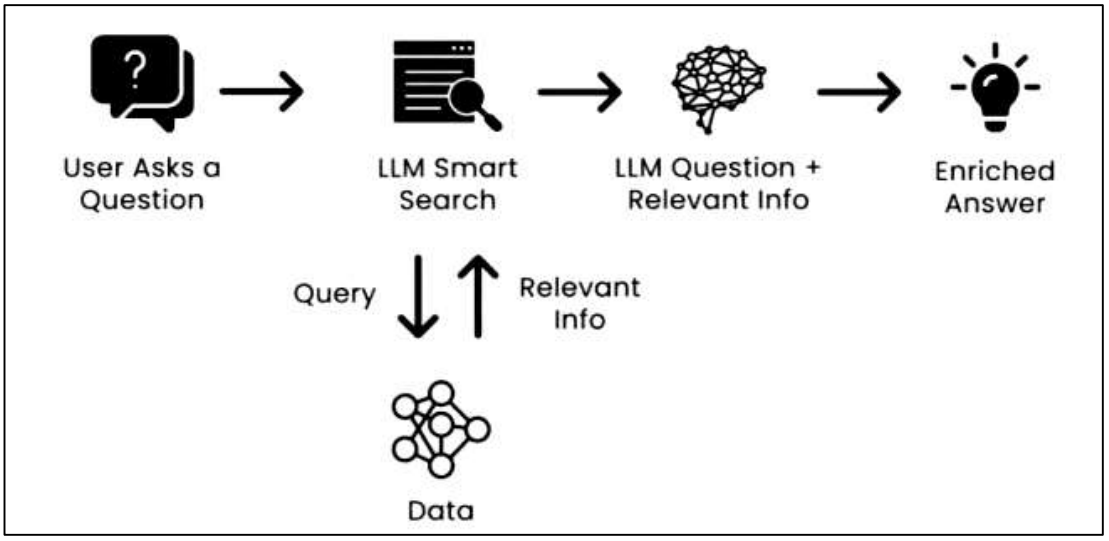

Рис.2.4. Загальний алгоритм логіки RAG

Головною перевагою RAG для наукового аналізу є гарантія актуальності та достовірності відповіді. Оскільки генератор базується на конкретних, щойно знайдених джерелах, система може оперувати найновішими даними з препринтів, щойно опублікованих стандартів чи патентних заявок. Це кардинально знижує ризик так званих "галюцинацій" LLM, коли модель генерує правдоподібну, але фактологічно неправильну інформацію [1, 13]. Крім того, RAG забезпечує прозорість і підзвітність: кожна відповідь може супроводжуватися посиланнями на вихідні документи (цитуванням), що відповідає академічним стандартам і дозволяє

дослідникy верифікувати інформацію. У контексті телекомунікацій це дозволяє будувати експертні системи для порівняння характеристик різних поколінь мереж (наприклад, 5G vs. 6G), аналізу сумісності обладнання з новими стандартами або оцінки патентних ландшафтів.

Реалізація ефективної RAG-системи стикається з низкою технічних викликів. Якість кінцевої відповіді прямо залежить від точності роботи ретрівера. Якщо він пропускає критично важливі документи або повертає нерелевантні фрагменти, генератор отримає спотворений контекст. Для покращення релевантності часто використовуються гібридні підходи, що поєднують щільний векторний пошук (наприклад, Dense Passage Retrieval) з лексичними методами (на кшталт BM25), які краще вловлюють точні співпадіння термінів. Іншим викликом є обмеження довжини контексту, який може обробити генеративна модель. Для роботи з довгими науковими документами необхідно застосовувати стратегії розумного вибору та агрегації найважливіших фрагментів. Незважаючи на це, RAG залишається найперспективнішим напрямком для створення інтелектуальних наукових асистентів, здатних вести предметну розмову, готувати огляди літератури та надавати експертні консультації на основі найсвіжіших даних.

2.2.3 Обробка різноформатних наукових джерел

Ефективність систем семантичного пошуку та RAG безпосередньо залежить від якості та структурованості вхідних даних. Однак наукові знання в телекомунікаціях представлені в різноманітних, часто погано структурованих форматах, основним завданням попередньої обробки є перетворення цих гетерогенних джерел в чистий, структурований текст, придатний для генерації ембедінгів. Найпоширенішими форматами є PDF, LaTeX та HTML, кожен з яких представляє свої унікальні виклики для парсингу.

PDF є найбільш поширеним форматом для фінального розповсюдження наукових статей, стандартів і патентів через свою стабільність та незалежність від платформи. Проте, будучи форматом, орієнтованим на візуальне представлення, він часто не містить явної семантичної розмітки. Видобування тексту, формул, таблиць та метаданих з PDF є складною задачею. Для її вирішення

використовуються спеціалізовані інструменти. GROBID (GeneRation Of BIbliographic Data) є потужним відкритим інструментом на основі машинного навчання, який аналізує PDF-файли наукових статей і видобуває структуровані метадані (автори, назва, бібліографія) та повний текст з розпізнаванням розділів, посилань і навіть простих математичних виразів. Для більш низькорівневого контролю можуть використовуватись бібліотеки, такі як PDFMiner або PyPDF2, але вони часто потребують додаткової обробки для відновлення логічної структури документа.

LaTeX, як мова розмітки, є основним інструментом для підготовки наукових рукописів, особливо в технічних дисциплінах. На відміну від PDF, вихідні файли LaTeX містять багату семантичну розмітку: чітко визначені розділи, математичні формули, цитування та бібліографічні записи. Це робить їх ідеальним джерелом для автоматичної обробки. Інструменти, такі як Pandoc або спеціалізовані парсери LaTeX, можуть конвертувати .tex-файли в структуровані формати (наприклад, XML, JATS), зберігаючи при цьому логічну структуру та математичний контент. Багато препринтів на arXiv доступні саме в форматі LaTeX, що відкриває можливість для якісного парсингу та подальшого аналізу.

HTML є основним форматом веб-контенту, включаючи онлайн-журнали, сторінки стандартів організацій (наприклад, IEEE, IETF) та інші наукові портали. HTML за своєю суттю є структурованим за допомогою тегів, що дозволяє відносно легко видобувати текст та його семантичну ієрархію (заголовки, абзаци, списки) за допомогою бібліотек, таких як BeautifulSoup або lxml для Python. Основним викликом тут може бути різноманітність і неконсистентність розмітки між різними веб-сайтами, що потребує написання специфічних парсерів або використання технік машинного навчання для розпізнавання шаблонів.

Загальним завданням для всіх форматів є приведення до єдиного внутрішнього представлення (наприклад, структурованого JSON або XML), яке зберігає ієрархію розділів, посилання, метадані та сам текст. Цей крок, відомий як нормалізація, є фундаментом для подальшої генерації якісних ембеддингів та ефективної роботи семантичного пошуку. Таким чином, побудова ефективного

пайплайну обробки документів для наукового ШІ-агента потребує комбінації цих інструментів. Ідеальний процес включає етап нормалізації, коли документи різних форматів приводяться до єдиного внутрішнього структурованого представлення (наприклад, JSON з полями для заголовка, абстракту, розділів, бібліографії). Це структуроване подання потім може бути розбите на менші, семантично зв'язні фрагменти (наприклад, на рівні абзаців або секцій), які індексуються в векторній базі даних. Подолання викликів, пов'язаних із втратою інформації при конвертації, особливо для складних таблиць, схем та формул, залишається активним напрямком досліджень, але сучасний інструментарій вже дозволяє створювати надійні системи для глибокого семантичного аналізу наукової літератури з телекомунікацій.

2.3 Інструменти для доступу до зовнішніх даних та API

2.3.1 Інтеграція з академічними API

Функціонування сучасних ШІ-агентів для наукового аналізу в телекомунікаціях неможливе без безперебійного доступу до актуальних, структурованих та авторитетних наукових даних. Цей доступ забезпечується через інтеграцію зі спеціалізованими академічними програмними інтерфейсами провідних відкритих репозиторіїв і бібліометричних систем. Серед ключових платформ, що формують основу для такого збору даних, є arXiv, PubMed та Crossref, а також новітні інструменти на кшталт OpenAlex, CORE та Semantic Scholar. Ефективне використання цих API дозволяє автоматизувати процеси моніторингу наукових трендів, формування тренувальних датасетів, побудови графів знань та забезпечення фактологічної основи для генеративних моделей.

Інтеграція з arXiv API забезпечує прямий доступ до величезного корпусу препринтів у галузях фізики, математики, інформатики та інженерії, включаючи чимало публікацій з телекомунікацій. Цей RESTful API не вимагає складної аутентифікації, що спрощує початкове впровадження, та повертає дані в структурованих форматах JSON або XML. Його ключова цінність полягає в можливості точної фільтрації за категоріями (наприклад, cs.NI для комп'ютерних мереж і комунікацій), авторами, датами та ключовими словами. Це дозволяє ШІ-агенту автоматично та регулярно оновлювати свою базу знань найновішими

дослідницькими результатами, відстежувати еволюцію конкретних технологічних напрямів, таких як розвиток протоколів 6G чи нових архітектур програмно-визначених мереж.

PubMed API (або Entrez E-utilities) відкриває доступ до світової біомедичної літератури, що стає надзвичайно важливим для міждисциплінарних досліджень на перетині телекомунікацій та охорони здоров'я. Цей інтерфейс підтримує складні запити з використанням MeSH-термінології (Medical Subject Headings), що дозволяє точно ідентифікувати роботи, пов'язані з телемедициною, носимими сенсорами, обробкою біомедичних сигналів та іншими суміжними темами. Наявність API-ключа дозволяє зняти обмеження на частоту запитів, що критично для масштабного збору даних. Інтеграція PubMed дає агенту змогу аналізувати не лише технічні аспекти, а й медичні вимоги та обмеження, що формує більш цілісне розуміння предметної області.

Crossref API є основним джерелом бібліографічних метаданих для наукових публікацій через систему цифрових ідентифікаторів об'єктів. Його основна сила — уніфікація та верифікація інформації з тисяч різних видавців. API дозволяє не лише отримувати базові метадані (автори, назва, джерело), але й будувати ланцюжки цитувань, що є фундаментом для аналізу наукового впливу, виявлення ключових статей та побудови карт знань (Knowledge Graphs). Для телекомунікаційного агента це означає можливість відстежувати, як конкретна ідея або технологія поширюється та розвивається в науковій літературі, ідентифікувати провідні дослідницькі центри та тенденції колаборацій.

Порівняльні характеристики цих API наведені в таблиці 2.3.

Таблиця 2.3

Порівняння ключових академічних API для інтеграції з ШІ-агентами

API / Платформа	**Основна сфера**	**Ключова перевага**	**Формат даних**	**Аутентифікація**	**Основне використання**
arXiv	STEM-дисципліни, препринти	Швидкий доступ до найновіших, ще не опублікованих результатів	JSON, XML, Atom	Не потрібна (обмеження за частотою)	Моніторинг інновацій, тренд-аналіз, поповнення бази препринтами
PubMed	Біомедицина, інженерія здоров'я	Глибока інтеграція з медичною термінологією (MeSH)	XML, JSON	API-ключ (для підвищених лімітів)	Міждисциплінарні дослідження (телемедицина, біосенсори)
Crossref	Усі дисципліни, метадані	Централізована база DOI, дані про цитування	JSON	Email у заголовку (рекомендовано)	Верифікація метаданих, аналіз цитувань, побудова графів знань
OpenAlex / CORE	Мультидисциплінарні, відкритий доступ	Агрегація відкритих даних з багатьох джерел, повні тексти	JSON, RDF	Варіюється (від відкритого до API-ключа)	Формування комплексних корпусів, семантичний пошук, RAG

Технічна інтеграція цих API в пайплайн ШІ-агента зазвичай реалізується за допомогою мови Python та бібліотек, таких як requests або aiohttp для асинхронних запитів. Для спрощення роботи існують спеціалізовані клієнтські бібліотеки (наприклад, arxiv для arXiv, biopython для PubMed, habanero для Crossref), які інкапсулюють логіку формування запитів і парсингу відповідей. Отримані дані

потім трансформуються, нормалізуються та зберігаються в придатному для подальшої обробки вигляді — наприклад, у векторній базі для семантичного пошуку або в реляційній базі для аналізу метаданих. Критично важливим є реалізація механізмів обробки помилок, дотримання політики лімітів запитів (rate limiting) та планування регулярних синхронізацій для підтримки актуальності даних.

2.3.2 Можливості веб-краулінгу для пошуку актуальних публікацій

Хоча академічні API надають структурований доступ до великих репозиторіїв, значна частина релевантної наукової інформації в телекомунікаціях розкидана по веб-сайтах конференцій, інституційних архівах, персональних сторінках дослідників та спеціалізованих порталах, які часто не надають публічних API. Для доступу до цих даних необхідне застосування веб-краулінгу (web crawling) та скрапінгу (web scraping) — автоматизованих технік збору та видобутку інформації з веб-сторінок. Ці методи дозволяють ШІ-агенту сформувати більш повний, актуальний та нішево орієнтований корпус документів, що є вирішальним для всебічного аналізу галузі.

Технологічний стек для веб-краулінгу включає низку потужних інструментів, кожен з яких вирішує певні завдання. Фреймворк Scrapy, написаний на Python, є одним з найпопулярніших рішень для створення масштабованих та ефективних краулерів. Він надає архітектуру для асинхронної обробки запитів, автоматичного дотримання правил robots.txt, обробки різних форматів виводу та легкої інтеграції в складні пайплайни обробки даних. Для більш простих завдань парсингу статичних HTML-сторінок часто використовується бібліотека BeautifulSoup, яка дозволяє швидко видобувати необхідні дані за допомогою CSS-селекторів або XPath-виразів. Однак сучасний веб все частіше побудований на динамічному JavaScript-контенті, який не завантажується при початковому запиті. Для роботи з такими сторінками застосовуються інструменти автоматизації браузера, такі як Selenium або Playwright. Вони дозволяють програмно імітувати дії користувача (кліки, прокручування, введення тексту), чекати завантаження динамічного контенту та потім видобувати необхідну інформацію. Це незамінно для збору даних

зі складних порталів конференцій або інтерактивних наукових баз.

Процес веб-краулінгу для наукового агента можна розділити на кілька етапів. Спочатку визначається список цільових джерел: офіційні сайти ключових конференцій (наприклад, IEEE GLOBECOM, ICC), архіви університетів з сильними телекомунікаційними програмами, блоги провідних дослідницьких груп, державні репозиторії звітів про фінансовані проєкти. Далі розробляються специфічні парсери для кожного типу джерела, які видобувають метадані (назва, автори, дата, абстракт, ключові слова) та посилання на повні тексти (PDF). На третьому етапі здійснюється завантаження повних текстів із відкритих джерел, після чого вони проходять процедуру нормалізації та інтегруються в загальну систему обробки документів.

Реалізація відповідального та ефективного краулінгу супроводжується низкою викликів. Технічні виклики включають необхідність адаптувати парсери до постійних змін у структурі веб-сторінок, подолання захисту від ботів (наприклад, CAPTCHA, Cloudflare), а також управління обмеженнями на швидкість запитів, щоб не перевантажувати сервери джерел. Юридичні та етичні аспекти є ще більш важливими [16]. Необхідно суворо дотримуватися правил, викладених у файлі robots.txt сайту, умов користування (Terms of Service) та авторських прав. Збір повних текстів статей зазвичай допустимий лише для матеріалів з відкритою ліцензією (Open Access). У випадку з комерційними видавцями (Elsevier, IEEE Xplore) можливий лише збір метаданих, якщо це дозволяється їхніми політиками. Для вирішення цих питань рекомендується використовувати легальні альтернативи, такі як офіційні API (де вони є), партнерські угоди з бібліотеками або зосередження на публічно доступних відкритих архівах.

Веб-краулінг, поєднаний з методами машинного навчання, може суттєво підвищити свою інтелектуальність. Наприклад, моделі класифікації текстів можуть автоматично фільтрувати зібрані документи за релевантністю до конкретних тем телекомунікацій. Моделі виявлення дублікатів можуть ідентифікувати одну й ту саму публікацію, завантажену з різних джерел. Таким чином, веб-краулінг перестає бути простим збором даних і стає інтелектуальним інструментом формування

якісного, спеціалізованого корпусу знань для автономного наукового агента в галузі телекомунікацій.

2.4 Існуючі open-source рішення для наукових досліджень

2.4.1 Огляд та порівняння проєктів типу ODR, GPT Researcher, ResearchGPT

У сучасному науковому середовищі, особливо в динамічній галузі телекомунікацій, все більшого значення набувають інструменти штучного інтелекту, здатні автоматизувати та прискорити дослідницькі процеси. Серед відкритих (open-source) рішень, що претендують на роль платформ для побудови спеціалізованих ШІ-агентів, виокремлюються три проєкти: Open Deep Research, GPT Researcher та ResearchGPT. Кожен із них пропонує власний підхід до архітектури, функціоналу та інтеграції з науковими джерелами, проте їхня придатність для комплексного наукового аналізу в телекомунікаціях суттєво різниться [2, 18]. Детальний порівняльний аналіз цих платформ дозволяє визначити найбільш перспективну основу для розробки власного агента, зорієнтованого на вимоги сучасних телекомунікаційних досліджень.

Open Deep Research позиціонується як універсальна платформа для глибокого наукового аналізу з використанням автономних агентів. Незважаючи на деяку обмеженість відкритої документації на поточний момент, заявлені архітектурні принципи та функціональні можливості роблять її надзвичайно привабливою для спеціалізованих завдань.

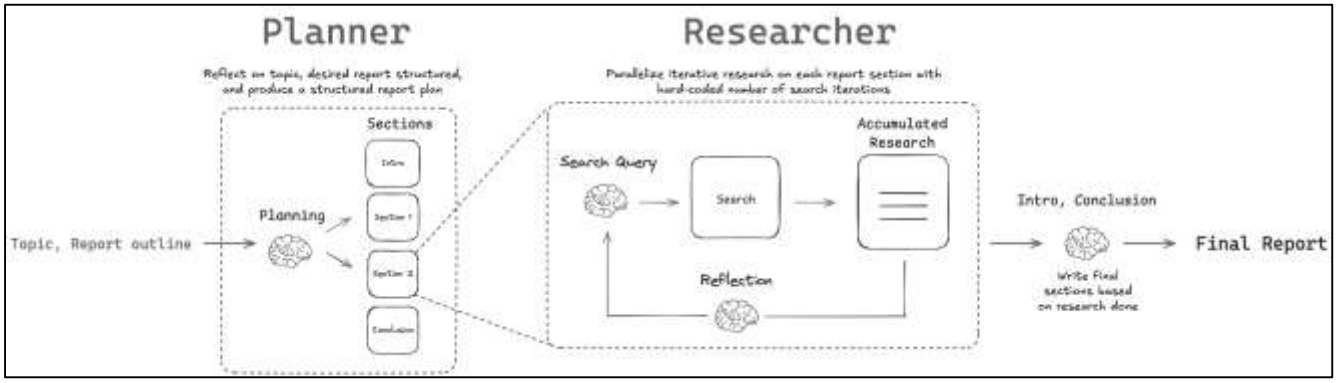

Рис.2.5. Блок-схема процесу аналізу Open Deep Research

Ключовою перевагою Open Deep Research є її модульна, гнучка архітектура,

побудована на принципах plug-and-play. Це дозволяє дослідникам адаптувати систему під конкретні потреби, додавати нові типи обробки даних та інтегрувати спеціалізовані інструменти. Платформа орієнтована на повний цикл наукового аналізу – від збору інформації з різноманітних джерел до глибокої семантичної обробки, критичної інтерпретації та формування висновків [2].

Важливою рисою є акцент на роботі з різноформатними науковими документами (PDF, LaTeX, HTML), що суттєво для телекомунікацій, де інформація представлена у вигляді статей, стандартів, патентів та технічних звітів. Крім того, Open Deep Research декларує підтримку інтеграції з ключовими академічними API (arXiv, Crossref, IEEE Xplore) та наявність власних механізмів веб-краулінгу для збору актуальних публікацій. Архітектурна концепція платформи передбачає підтримку контейнеризації та мікросервісного підходу, що забезпечує високу масштабованість та можливість обробки великих обсягів даних, що критично важливо для аналізу масштабних телекомунікаційних мереж та потоків даних. Таким чином, Open Deep Research пропонує комплексне, адаптивне середовище, здатне стати основою для створення потужного наукового агента, здатного не лише знаходити інформацію, але й проводити її глибинний причинно-наслідковий аналіз.

GPT Researcher є популярним відкритим проєктом, орієнтованим на автоматизацію літературного огляду та генерацію дослідницьких звітів. Його архітектура також є модульною та базується на використанні великих мовних моделей (LLM), таких як GPT.

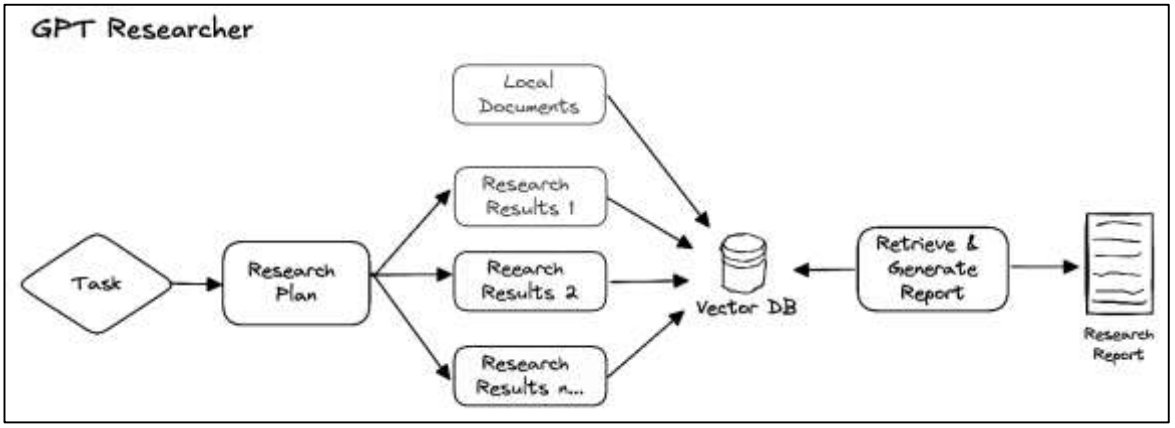

Рис.2.6. Блок-схема процесу аналізу GPT Researcher

Він пропонує готові пайплайни для пошуку інформації через інтеграцію з

численними академічними API (Semantic Scholar, arXiv, Europe PMC тощо) та веб-краулінг. Система дозволяє автоматизувати процеси формування запитів, збору релевантних документів, їх фільтрації та реферування. GPT Researcher має активну спільноту, добре документований код та підтримує розгортання за допомогою сучасних інструментів (Docker, Celery), що спрощує його використання. Однак, його функціонал більше зосереджений на етапах пошуку та попереднього аналізу, тоді як можливості глибокої семантичної обробки, побудови складних логічних ланцюжків та критичної інтерпретації, необхідних для фундаментальних досліджень у телекомунікаціях, можуть бути обмеженими. Крім того, архітектура GPT Researcher може вимагати значних зусиль для адаптації під вузькоспеціалізовані задачі обробки телекомунікаційних даних (наприклад, сигналів, лог-файлів, специфікацій протоколів) [18].

ResearchGPT є іншим проєктом, що використовує LLM для автоматизації досліджень. Проте, на відміну від перших двох, він має менш розвинену екосистему, обмежену документацію та невелику спільноту підтримки. Його функціональні можливості здебільшого зосереджені на базовому пошуку та синтезі текстів на основі обмеженого кола джерел. Архітектура ResearchGPT є менш гнучкою та масштабованою, що обмежує її застосування у складних, ресурсомістких завданнях телекомунікаційної науки, де потрібна обробка великих масивів структурованих і неструктурованих даних, інтеграція з галузевими стандартами та високий рівень адаптації.

Для наочної порівняльної оцінки цих платформ у контексті потреб наукового аналізу в телекомунікаціях наведемо таблицю 2.4.

Таблиця 2.4

Порівняльний аналіз open-source платформ для наукових ШІ-агентів

Критерій	Open Deep Research	GPT Researcher	ResearchGPT
Архітектурна гнучкість	Висока: модульна, plug-and-play архітектура, орієнтована на адаптацію та розширення.	Висока: модульна структура з чіткими пайплайнами.	Середня: обмежена модульність, менш гнучка структура.
Масштабованість	Висока: підтримка мікросервісів, контейнеризації, обробка великих даних.	Висока: можливість паралельної обробки, інтеграція з Celery для розподілених задач.	Середня: обробка середніх обсягів даних, можливі обмеження при зростанні.
Робота з науковими документами	Комплексна: підтримка PDF, LaTeX, HTML; семантична індексація, глибока обробка.	Базова: підтримка PDF та HTML, орієнтація на реферування та пошук.	Обмежена: базові функції парсингу та обробки текстів.
Інтеграція з зовнішніми API	Широка: заявлена підтримка arXiv, Crossref, IEEE Xplore та інших академічних джерел.	Широка: вбудована підтримка багатьох академічних API (Semantic Scholar, arXiv тощо).	Обмежена: підтримка основних API, можлива потреба в доопрацюванні.
Веб-краулінг	Є: власні механізми для збору актуальних публікацій.	Є: інтегровані інструменти для веб-скрапінгу.	Обмежена: базові можливості краулінгу.
Глибина аналізу	Висока: орієнтація на причинно-наслідковий аналіз, критичну інтерпретацію, побудову логічних ланцюжків.	Середня: орієнтація на пошук, агрегацію та реферування; глибина аналізу залежить від LLM.	Низька: базовий синтез інформації на основі LLM.

Продовження таблиці 2.4

Критерій	Open Deep Research	GPT Researcher	ResearchGPT
Адаптація до телекомунікацій	Висока потенційно: модульність дозволяє додавати спеціалізовані обробники даних, інтегрувати галузеві стандарти.	Середня: можлива адаптація через модифікацію пайплайнів, але потребує зусиль.	Низька: обмежені можливості для адаптації під вузькі галузеві задачі.
Документація та спільнота	Обмежена наразі: потребує подальшого розвитку та відкриття.	Висока: добре документований код, активна спільнота, регулярні оновлення.	Низька: обмежена документація, невелика спільнота.
Етична та правова відповідність	Акцент на дотриманні robots.txt, авторських прав та етичних норм збору даних.	Враховує правила користування API та веб-джерел.	Інформація обмежена.

2.4.2 Критерії вибору базової архітектури для власної розробки

На основі проведеного порівняльного аналізу існуючих open-source платформ для наукового ШІ-аналізу, для подальшої розробки власного спеціалізованого агента в галузі телекомунікацій необхідно визначити чіткі критерії вибору базової архітектури. Ці критерії мають відображати як загальні вимоги до сучасних автономних інтелектуальних систем, так і специфічні потреби наукових досліджень у динамічній та технічно складній сфері телекомунікацій. Вибір оптимальної основи є критичним етапом, оскільки він визначає масштабованість, гнучкість та ефективність майбутнього рішення.

Основними критеріями вибору базової архітектури є:

1) Модульність та архітектурна гнучкість. Архітектура повинна ґрунтуватися на принципах компонентної розробки, що дозволяє незалежно додавати, замінювати або модифікувати окремі функціональні блоки. Це критично

важливо для адаптації системи під вузькоспеціалізовані задачі телекомунікацій, такі як обробка специфічних протоколів, аналіз сигналів або інтеграція з обладнанням;

2) Масштабованість та продуктивність. Платформа повинна підтримувати обробку великих обсягів структурованих і неструктурованих даних, характерних для телекомунікацій (трафік, логи, наукові корпуси). Бажаною є підтримка розподілених обчислень, контейнеризації (Docker, Kubernetes) та асинхронної обробки запитів;
3) Глибина аналітичної обробки. Архітектура має сприяти реалізації не лише пошукових, але й аналітичних функцій, включаючи причинно-наслідковий аналіз, семантичне розуміння контексту, критичну оцінку джерел та генерацію науково обґрунтованих гіпотез – тобто повноцінну підтримку моделі RAO;
4) Інтеграційна здатність. Критерій включає легку інтеграцію зі зовнішніми академічними API (arXiv, IEEE Xplore, Crossref), векторними базами даних (ChromaDB, Weaviate) та інструментами веб-краулінгу. Це забезпечує автономність агента в пошуку та оновленні знань;
5) Адаптація до предметної галузі. Архітектура повинна дозволяти легко впроваджувати доменні знання, термінологію та спеціалізовані моделі обробки даних, характерні для телекомунікацій (наприклад, обробку параметрів мережі, стандартів зв'язку);
6) Відкритість та підтримка спільноти. Наявність відкритого вихідного коду, ліцензії, що дозволяє модифікації, активної спільноти розробників та користувачів, а також якісної документації є ключовими факторами для тривалого розвитку та підтримки власної розробки;
7) Відповідність принципам відтворюваності досліджень. Архітектура має сприяти прозорому логуванню дій агента, збереженню контексту експериментів та можливості відтворення отриманих результатів, що є основним вимогам сучасної науки.

Порівняння розглянутих платформ за цими критеріями (Таблиця 2.4)

однозначно вказує на Open Deep Research як на найбільш відповідну основу для власної розробки. Незважаючи на поточний стан обмеженої публічної документації, концептуальні переваги цієї архітектури є вирішальними:

- Незрівнянна гнучкість та модульність. Архітектура «plug-and-play» Open Deep Research дозволяє будувати агента саме під конкретні дослідницькі потоки в телекомунікаціях. Можливість легкого впровадження спеціальних модулів для обробки телекомунікаційних даних, інтеграції з симуляторами мереж (наприклад, NS-3) або обладнанням робить її ідеальним фундаментом.
- Орієнтація на глибину аналізу. На відміну від GPT Researcher, який зосереджений переважно на пошуку та агрегації, Open Deep Research концептуально націлена на реалізацію повного циклу наукового мислення (Reasoning). Це дозволить розробити агента, здатного не просто знаходити статті про, наприклад, алгоритми маршрутизації в 5G, але й аналізувати їх ефективність у різних умовах, формувати власні гіпотези щодо оптимізації та планувати віртуальні експерименти для їх перевірки.
- Вбудована підтримка критичної інфраструктури. Заявлена підтримка роботи з різноформатними джерелами, векторними базами даних та академічними API напряму відповідає технологічному ланцюгу, описаному в підрозділах 2.2 та 2.3. Це знімає необхідність розробки базової інфраструктури «з нуля» та дозволяє сконцентруватися на галузевій спеціалізації.
- Потенціал для подолання поточних обмежень. Відкритість коду та модульна природа Open Deep Research дозволяють компенсувати поточний брак документації шляхом активного аналізу вихідного коду та розвитку власних, добре документованих модулів. Таким чином, обрана архітектура не є «чорною скринькою», а надає свободу для її адаптації та вдосконалення відповідно до академічних стандартів.

Отже, вибір Open Deep Research як базової архітектури для власної розробки спеціалізованого ШІ-агента для телекомунікацій є обґрунтованим стратегічним рішенням. Ця платформа надає оптимальний баланс між потужною, але гнучкою

архітектурою, орієнтацією на глибокий науковий аналіз та можливістю повної адаптації під специфіку предметної галузі. Подальші етапи розробки будуть полягати в детальному вивченні її вихідного коду, розробці та інтеграції спеціалізованих модулів для телекомунікацій, що дозволить реалізувати потужний інструмент для автоматизації наукових досліджень у цій галузі.

Висновки

Аналіз архітектурних підходів та технологічних рішень для побудови спеціалізованих ШІ-агентів у сфері наукового аналізу демонструє комплексну взаємозалежність між теоретичними парадигмами, програмними інструментами та предметними галузевими вимогами. Модель «Мислення-Дії-Спостереження» виявляється концептуальним стрижнем, що забезпечує циклічність та адаптивність, необхідні для імітації наукового методу в автономних системах. Її практична реалізація знаходить своє втілення через низку спеціалізованих фреймворків, кожен з яких акцентує увагу на різних аспектах ланцюжка обробки даних – від динамічного планування та оркестрації до індексно-орієнтованого пошуку та пайплайнової обробки. Ця розмаїтість підходів підкреслює відсутність універсального рішення, а необхідність вибору технології продиктована специфікою дослідницького завдання, зокрема в технічно складній та динамічній галузі телекомунікацій, де поєднуються потреба в обробці великих обсягів структурованих даних мереж та неструктурованої наукової літератури.

Ключовим компонентом, що забезпечує інтелектуальну основу для таких агентів, є технології семантичного пошуку та роботи з документами. Перехід від лексичного до векторного пошуку, реалізований через системи на кшталт ChromaDB, Weaviate та Qdrant, принципово змінює можливості навігації в наукових корпусах, дозволяючи виявляти глибинні концептуальні зв'язки незалежно від конкретної термінології. Ця можливість посилюється архітектурою RAG, яка елегантно вирішує проблему застарівання знань великих мовних моделей шляхом їх динамічного забезпечення актуальним контекстом, що критично важливо для швидкоєволюціонуючих телекомунікаційних технологій. Однак ефективність цих систем повністю залежить від якості попередньої обробки

різноформатних джерел, що вимагає створення надійних пайплайнів конвертації PDF, LaTeX та HTML документів в єдине структуроване подання, здатне зберегти наукову семантику.

Функціонування автономного агента немислиме без сталого механізму поповнення його бази знань, що досягається через інтеграцію з академічними API та методами веб-краулінгу. Синергія між структурованими джерелами типу arXiv або Crossref та скрапінгом розрізнених веб-ресурсів дозволяє сформувати всебічний і актуальний інформаційний корпус. При цьому технічна реалізація цих процесів нерозривно пов'язана з дотриманням правових та етичних норм, що накладає додаткові вимоги до архітектури системи управління даними. Порівняльний аналіз існуючих open-source платформ свідчить, що їх придатність для фундаментальних досліджень суттєво варіюється. Якщо GPT Researcher ефективно вирішує задачі автоматизації літературного огляду, то Open Deep Research, завдяки своїй модульній архітектурі, орієнтації на глибинний причинно-наслідковий аналіз та заявленій підтримці повного циклу роботи з даними, прогнозується як найбільш перспективна основа для побудови спеціалізованого агента. Її концепція дозволяє інтегрувати розглянуті технології – від фреймворків реалізації RAO до векторних баз і інструментів збору даних – в єдину систему, здатну не тільки знаходити інформацію, але й проводити власне наукове мислення, формулюючи та перевіряючи гіпотези в контексті телекомунікацій.

Таким чином, узагальнення розглянутих технологічних ланок вказує на формування цілісного архітектурного шаблону для наукових ШІ-агентів. Цей шаблон поєднує циклічну логіку мислення з потужними інструментами семантичного доступу до знань та динамічного оновлення контексту. Успішна реалізація такої системи в галузі телекомунікацій вимагатиме глибокої галузевої спеціалізації окремих модулів, але саме такий інтегрований підхід відкриває шлях до створення дійсно автономних інтелектуальних систем, здатних прискорювати генерацію нових знань, автоматизувати складні експериментальні протоколи та підвищувати ефективність досліджень у умовах стрімкого зростання обсягів інформації та технічної складності сучасних комунікаційних технологій.

РОЗДІЛ 3

РОЗРОБКА ТА НАЛАШТУВАННЯ ДОСЛІДНИЦЬКОГО ШІ-АСИСТЕНТА

3.1 Архітектура та компоненти системи

У контексті автоматизації наукового аналізу, вибір інструментарію є критичним завданням. Серед різноманіття підходів — від розробки власних систем з нуля до використання комерційних платформ — особливе місце займають відкриті (open-source) фреймворки, що поєднують гнучкість, прозорість архітектури та сучасні можливості. Саме таким рішенням є ODR, спеціалізований фреймворк, розроблений на базі екосистеми LangChain [2].

ODR позиціонується як повноцінна платформа для автоматизації складного науково-дослідницького workflow. Його основним призначенням є імітація структурованого процесу, який виконує досвідчений дослідник: від формулювання проблеми та планування до пошуку інформації, її критичного аналізу, синтезу та генерації когерентного звіту. Філософія проекту полягає у наданні науковцям та інженерам конфігурованої, але гнучкої системи, яка поєднує потужність сучасних LLM з можливістю інтеграції власних моделей, інструментів пошуку та серверів контексту (MCP).

Ключовою перевагою ODR є його відкритий характер. Це дозволяє:

- Повний огляд архітектури та логіки: Дослідник може детально вивчити код, зрозуміти механізми прийняття рішень агентом, модифікувати окремі компоненти під конкретні наукові потреби (наприклад, додати спеціалізовані бази даних публікацій або аналітичні інструменти для телекомунікаційних стандартів);
- Порівнянна продуктивність: Незважаючи на статус open-source, незалежні бенчмарки демонструють, що продуктивність ODR у задачах глибокого автономного дослідження є порівнянною з комерційними закритими аналогами. Це робить його життєздатним вибором для академічних установ з обмеженим бюджетом;
- Ідеальний фундамент для академічних та прикладних задач: Система

задумана як основа, що може адаптуватися під різноманітні сценарії: від проведення систематичних літературних оглядів (systematic literature review) до моніторингу технологічних трендів, конкурентного аналізу та підготовки технічних звітів у динамічній галузі телекомунікацій.

ODR виступає не просто як "чорний ящик", а як модульна та розширювана лабораторія, яка дає досліднику контроль над процесом автоматизації, що є критично важливим для наукової строгості та відтворюваності результатів. Серцем ODR є комбінація двох потужних фреймворків: LangChain та LangGraph. Ця комбінація дозволяє перейти від лінійних сценаріїв взаємодії з LLM до побудови складних, станомістких, багатоагентних систем, здатних до адаптивного планування та виконання.

LangChain виступає базовим каркасом, який надає:

- Стандартизовані інтерфейси: Абстрагує взаємодію з різними LLM (OpenAI, Anthropic, локальні моделі тощо), роблячи систему незалежною від постачальника;
- Концепцію "Інструментів" (Tools): Дозволяє легко інтегрувати зовнішні сервіси та API (наприклад, пошукові системи, бази даних, калькулятори) як інструменти, якими агент може користуватися. У ODR це, перш за все, інструменти веб-пошуку;
- Управління контекстом: Надає шаблони для роботи з пам'яттю агента, промптами та цепочками викликів, що є основою для структурованої взаємодії;
- "З'єднувальну тканину" між різними компонентами системи.

Якщо LangChain надає компоненти, то LangGraph надає механізми для їх складної координації. Він дозволяє моделювати роботу системи як орієнтованого графа, де:

- Вузли (Nodes): Представляють окремі кроки або дії в процесі. Це може бути виклик LLM, виконання інструменту (пошук), перевірка умови, об'єднання даних. У ODR ключовими вузлами є етапи `Scope`, `Research` та `Write`, а

також вузли для підагентів (sub-agents);

– Ребра (Edges): Визначають потік виконання між вузлами. Вони можуть бути умовними, тобто наступний крок вибирається динамічно на основі результату попереднього вузла. Це імітує людську логіку прийняття рішень: "якщо знайдено достатньо джерел по темі А, перейти до аналізу; якщо ні — уточнити пошуковий запит";
– Стан (State): Центральна структура даних, яка передається між вузлами та містить всю контекстну інформацію про поточне завдання: оригінальний запит користувача, сформований бриф, проміжні результати пошуку, висновки агентів, чернетку звіту тощо. Граф маніпулює цим станом, поступово збагачуючи його.

Ключова перевага такої архітектури для наукового аналізу — адаптивність. На відміну від лінійного скрипта, граф ODR може приймати рішення на льоту: наприклад, на етапі `Research`, якщо бриф містить дві незалежні підтеми ("протокол 5G NR" та "протокол Wi-Fi 6E"), супервайзер-вузел може вирішити створити паралельні гілки виконання для двох підагентів. Підагент може в циклі "пошук -> аналіз знайдених джерел -> оцінка достатності інформації" повторювати пошук з уточненими запитами, поки не буде задоволено критерії якості, визначені в брифі. LangGraph ефективно управляє паралельною роботою кількох підагентів, ізолюючи їх контексти для запобігання "змішуванню" інформації, а потім артегує їх результати на етапі синтезу.

Однією з найважливіших архітектурних відмінностей ODR від класичних RAG-систем є його фундаментальна орієнтація на пошук інформації в реальному часі. Традиційні системи для наукового аналізу часто базуються на попередньо підготовлених векторних базах даних, що містять індексовані наукові статті (наприклад, з arXiv, PubMed, IEEE Xplore) [12]. Хоча такий підхід забезпечує швидкий доступ до певного корпусу літератури, він має ключові недоліки для динамічних галузей, таких як телекомунікації:

– Статичність та затримка оновлення: База стає "знімком" минулого. Нові

публікації, стандарти, звіти про конференції або технічні доповіді, що з'являються щодня, відсутні до наступного циклу індексації;

– Обмежений охоплення: Система може працювати тільки з попередньо заданими джерелами. Вона не здатна знайти актуальні блоги технологічних компаній, офіційну документацію на сайтах розробників (наприклад, 3GPP, IETF), прес-релізи або обговорення в професійних спільнотах;
– Відсутність контекстуальної актуальності: Для багатьох практичних завдань (конкурентний аналіз, моніторинг трендів) ключову роль грає саме найновіша, часто неакадемічна інформація.

Щоб подолати ці обмеження, ODR за замовчуванням інтегрований з Tavily Search API – спеціалізованою пошуковою системою, оптимізованою для роботи з LLM. Tavily це API, спеціально розроблений для задач, де LLM потрібен доступ до актуальних, фактичних та релевантних даних з інтернету. Tavily не обмежується одним пошуковим рушієм. Він паралельно опитує та агрегує результати з різноманітних публічних та преміум-джерел, включаючи наукові репозиторії, новинні сайти, офіційну технічну документацію, форуми та блоги.

Алгоритми Tavily аналізують отримані результати на основі кількох критеріїв, критично важливих для наукового дослідження:

– Авторитетність джерела: Пріоритет віддається академічним установам, офіційним стандартизаційним органам (наприклад, ITU, ETSI), рецензованим журналам та технічним звітам відомих компаній;
– Свіжість: Новіші матеріали отримують вищий пріоритет, що критично для тем, що швидко розвиваються;
– Релевантність контексту LLM: Система фільтрує сторінки з переважно рекламним контентом, низькоякісні сайти та нерелевантну інформацію, "підготовлюючи" чисті, фактичні дані для подальшого аналізу моделлю;
– Об'єктивність: Намагається надавати збалансовану вибірку, уникаючи надмірного упередження до одного джерела.

Tavily API повертає не просто список посилань, а структурований JSON-

об'єкт, який зазвичай містить для кожного результату: заголовок, URL, стислий зміст (snippet), дату публікації та оцінку релевантності. Цей формат ідеально підходить для подальшої обробки агентом ODR.

Архітектура ODR з використанням Tavily вирішує проблему "ручного кураторства джерел" через абстракцію та автоматизацію. Замість того, щоб розробнику або досліднику потрібно було окремо інтегрувати API arXiv, потім IEEE Xplore, потім пошук Google Scholar і налаштовувати парсери для кожного, ODR використовує Tavily як універсальний "шлюз". Користувач формує науковий запит природною мовою, а Tavily відповідає за те, щоб знайти найкращі доступні джерела для цього запиту серед усієї мережі, включаючи ті самі arXiv та IEEE Xplore, якщо вони релевантні.

Для кожного запиту підбірка джерел формується динамічно. Якщо тема стосується останніх досліджень з машинного навчання для оптимізації мереж – система автоматично знайде актуальні препринти на arXiv. Якщо ж запит стосується нового ринкового звіту про впровадження 5G – пріоритет буде відданий авторитетним аналітичним агентствам (наприклад, Gartner, Ericsson Mobility Report). Дослідник не повинен заздалегідь знати, в якій саме базі даних може знаходитися потрібна інформація. Це особливо важливо для міждисциплінарних досліджень на стику телекомунікацій та інших галузей (наприклад, медицини або енергетики), де корисна інформація може розсіяна по десятках спеціалізованих джерел.

Важливою перевагою ODR є його глибока конфігуровність, що починається з вибору фундаментальної мовної моделі. Система підтримує роботу з різними LLM, включаючи Anthropic Claude, OpenAI GPT серії та відкриті моделі. У рамках даного дослідження була обрана модель gpt-4.1, що обумовлено її розширеними можливостями в розумінні складних технічних текстів, логічному міркуванні та генерації структурованих відповідей. Критичним параметром для наукового аналізу є розмір контекстного вікна. Використання моделі з великим контекстом дозволяє системі одночасно оперувати об'ємним брифом, численними проміжними висновками підагентів та повнотекстовими фрагментами знайдених джерел, що

суттєво підвищує глибину та зв'язність фінального синтезу.

Крім того, система надає доступ до низькорівневих механізмів налаштування. Дослідник може редагувати сам граф виконання, додаючи нові вузли-інструменти (наприклад, для спеціалізованої математичної обробки даних), змінюючи умовні переходи або впроваджуючи додаткові цикли перевірки якості. Кастомізація промптів агентів дозволяє тонко налаштувати їх поведінку: задати стиль аналізу (наприклад, з критичним або порівняльним ухилом), формат представлення висновків, глибину вивчення кожного джерела та правила цитування. Ця гнучкість перетворює ODR з готового рішення в адаптивну платформу, яку можна точно настроїти під специфічні методологічні вимоги конкретного наукового дослідження в галузі телекомунікацій.

Сучасні завдання автоматизації наукового аналізу, особливо в динамічних галузях, таких як телекомунікації, вимагають переходу від лінійних, детермінованих скриптів до систем, здатних до адаптивного планування, паралельного виконання та інтелектуального синтезу інформації. Архітектура таких систем повинна імітувати не лише окремі дії дослідника (пошук, читання), але й його високу когнітивну функцію — здатність структурувати проблему, розподіляти увагу між її аспектами та інтегрувати розрізнені знання в цілісну модель. ODR реалізує цей перехід через архітектуру, засновану на концепції оркестрованого багатоагентного графа зі збереженням стану. Ця парадигма дозволяє системі не пасивно обробляти запит, а активно керувати процесом його дослідження, приймаючи динамічні рішення про розгалуження логіки виконання на основі проміжних результатів. Загальний огляд взаємодії компонентів розкриває, як абстрактні фази наукового мислення (планування, збір даних, аналіз, синтез) трансформуються у формальну послідовність взаємопов'язаних програмних модулів, координація яких забезпечує досягнення когерентного кінцевого результату.

Фундаментальна логіка роботи ODR може бути візуалізована у вигляді високорівневої діаграми, що абстрагується від деталей реалізації та фокусується на логічних блоках та потоці даних між ними. Ця діаграма служить концептуальною

картою для розуміння того, як система трансформує неформальний запит користувача у структурований науковий звіт.

Рис.3.1. Загальний робочий цикл системи (макро-рівень)

На макро-рівні система реалізує послідовність трьох чітко розмежованих, але інформаційно пов'язаних фаз, що відповідають етапам методологічного дослідження.

1) Фаза Scoping (Уточнення та формування брифу): Система приймає початковий, часто нечіткий запит користувача. Замість негайного переходу до пошуку, вона ініціює процес уточнення, спрямований на розкриття контексту, цілей та критеріїв успіху. Результатом цієї фази є сформований дослідницький бриф — компактний, структурований документ, який слугує основним технічним завданням для всіх наступних компонентів. Цей крок реалізує принцип "контекстної компресії", замінюючи повну історію діалогу стислим планом, що зменшує накладні витрати на передачу інформації та підвищує точність наступних етапів;
2) Фаза Research (Планування та паралельний пошук): Отримавши бриф, система переходить до фази активного дослідження. Ця фаза є найбільш складною з точки зору оркестрації. Спочатку відбувається внутрішнє планування, під час якого центральний агент-супервайзер аналізує бриф і

визначає оптимальну стратегію його виконання. Для складних, багатоаспектних тем супервайзер приймає рішення про декомпозицію завдання на незалежні або слабкозв'язані підтеми. Кожній такій підтемі призначається спеціалізований підагент (sub-agent), який запускається паралельно. Кожен підагент виконує власний ітеративний цикл "пошук-аналіз", використовуючи зовнішні інструменти (наприклад, Tavily Search API). Проміжні результати кожного агента ізолюються та стискаються, що запобігає "смішенню" контекстів та неконтрольованому зростанню використання токенів моделі;

3) Фаза Synthesis (Агрегація та генерація звіту): Після завершення паралельної роботи всіх підагентів система вступає у заключну фазу синтезу. На цьому етапі стислі, структуровані висновки по кожній підтемі, супроводжені посиланнями на джерела, агрегуються разом із оригінальним брифом. Спеціалізований компонент (часто окремий виклик потужної LLM) відповідає за генерацію фінального звіту. Його завдання — не механічно склеїти тексти, а створити новий, когерентний наратив, що логічно інтегрує знання з усіх джерел у відповідності до початкових цілей, усуваючи повторення та суперечності. Результатом фази Synthesis є готовий, всебічний документ, що передається користувачеві як фінальна відповідь.

Конкретна реалізація описаного трифазного циклу в ODR базується на фреймворку LangGraph, який надає формальну модель для побудови складних workflow у вигляді орієнтованих графів. У цій моделі кожна фаза та підпроцес реалізуються як вузли (nodes), а логіка переходу між ними — як ребра (edges).

Центральна роль стану (State Object). Взаємодія між усіма компонентами системи відбувається через єдину, централізовану структуру даних — об'єкт стану. Цей об'єкт, типізований за допомогою Pydantic, є контейнером, що передається від вузла до вузла графа. Він містить усю динамічну інформацію про поточне виконання: оригінальний запит користувача, історію діалогу, сформований дослідницький бриф, список делегованих підзавдань, масив проміжних результатів пошуку та аналізу, чернетки висновків підагентів і, нарешті, готовий звіт. Така

архітектура забезпечує прозорість потоку даних і робить стан системи спостережуваним на будь-якому етапі, що є критичним для налагодження та аналізу.

ODR успадковує та розвиває парадигму динамічної оркестрації, де центральний агент-супервайзер виступає активним координатором. На відміну від жорстких workflow з попередньо визначеними умовними переходами, в цій моделі супервайзер, керований LLM, аналізує поточний стан і приймає рішення про подальші дії: чи продовжити уточнення, розпочати дослідження, делегувати підзавдання або перейти до синтезу. Його ключовим інструментом делегації є не простий виклик функції, а механізм ручного використання інструментів (manual tool-use), який ініціює запуск цілого нового підграфа LangGraph для кожного підагента. Це забезпечує справжню паралельність та ізоляцію контексту, оскільки кожен підагент виконується як незалежний процес зі своїм циклом прийняття рішень.

Взаємодія компонентів також опосередкована модульною системою конфігурації. Перед початком виконання графа формується об'єкт конфігурації, що агрегує налаштування з різних рівнів: значення за замовчуванням, змінні середовища та параметри конкретного запиту. Ця конфігурація включає вибір моделей LLM для різних ролей (наприклад, для супервайзера та підагентів можуть використовуватись різні моделі), налаштування пошукового бекенду, правила використання MCP-серверів, політики щодо кількості ітерацій пошуку тощо. Кожен компонент, отримуючи доступ до стану, читає необхідні йому конфігураційні параметри, що забезпечує гнучкість системи без необхідності зміни коду ядра графа.

Для глибокого розуміння принципів роботи системи ODR необхідно перейти від високорівневої, фазової моделі до детального аналізу програмних компонентів та механізмів їх взаємодії. Наступна структурна діаграма візуалізує ключові сутності системи та потоки даних між ними, розкриваючи внутрішню організацію кожної фази.

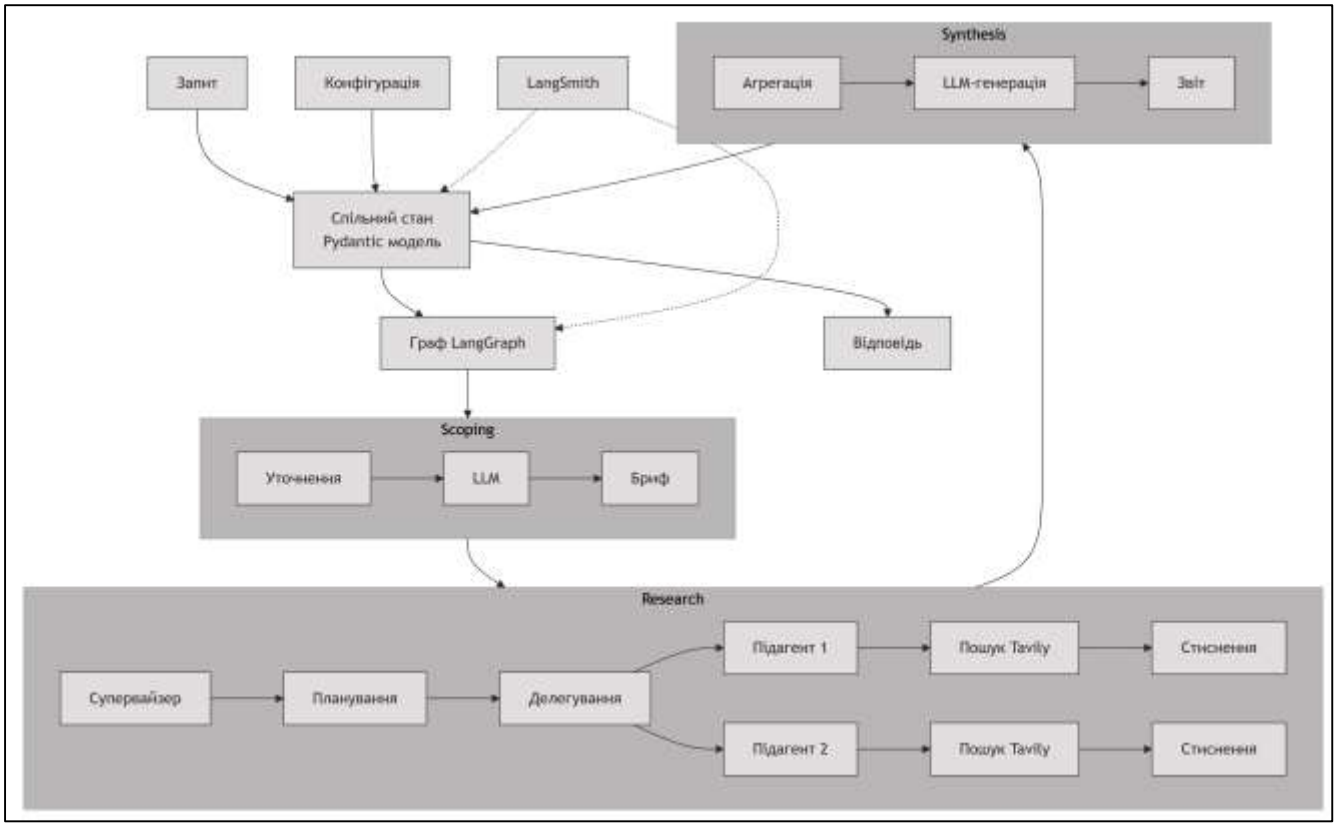

Рис.3.2. Деталізована архітектура взаємодії компонентів

Всі процеси в системі обертаються навколо центральної структури даних — спільного стану (Shared State), реалізованого як Pydantic-модель. Цей об'єкт є єдиним джерелом істини протягом життєвого циклу обробки запиту. Він ініціалізується в момент надходження запиту користувача та поступово збагачується даними на кожному наступному етапі. Стан містить як вхідні параметри (оригінальний запит, історію діалогу), так і всі проміжні артефакти: сформований бриф, список дослідницьких завдань, результати пошуку, стислі висновки підагентів та фінальний звіт. Така архітектура забезпечує детермінованість та спостережуваність: будь-який компонент системи, від супервайзера до підагента, отримує контекст виключно через цю структуру і

записує свої результати назад у неї. Паралельно зі станом формується об'єкт конфігурації, який агрегує налаштування з різних рівнів (за замовчуванням, змінні середовища, параметри запиту). Ця конфігурація, що включає вибір моделей LLM, пошукових бекендів та правил виконання, ін'єктується в стан, забезпечуючи всім компонентам доступ до єдиних правил гри без необхідності окремого зчитування параметрів.

Граф LangGraph виступає виконавчим двигуном, який послідовно активує вузли, репрезентовані підграфами Scoping, Research та Synthesis:

- Підграф Scoping (Уточнення та формування брифу). Цей етап розпочинається з вузла Уточнення, який аналізує початковий запит на предмет неоднозначностей або недостатньої конкретики. Потім управління передається LLM-вузлу, основним завданням якого є проведення структурованої бесіди (якщо це передбачено конфігурацією) для виявлення справжніх потреб користувача, контексту, часових рамок та очікуваного формату відповіді. Результатом цієї інтеракції є генерація формалізованого Дослідницького брифу (Research Brief). Цей документ, сформований як стислий, але вичерпний план, стає основним директивним документом для наступних етапів, виконуючи роль технічного завдання. Його створення є першим етапом контекстної компресії, що значно зменшує обсяг інформації, який потрібно передавати далі;
- Підграф Research (Дослідження: планування, делегування, паралельне виконання). Найскладніший за оркестрацією підграф. Вхідним вузлом виступає Супервайзер — центральний агент, відповідальний за стратегічне планування. Отримавши бриф, супервайзер виконує вузол Планування, під час якого аналізує тему на предмет можливості декомпозиції. Для складних запитів (наприклад, "Порівняння архітектури Open RAN та віртуалізації мережевих функцій") супервайзер розбиває тему на незалежні логічні блоки. Далі, на етапі Делегування, для кожного визначеного блоку створюється окремий Підагент (Sub-agent). Важливо відзначити, що делегування відбувається не через простий виклик функції, а через механізм запуску

нового, ізольованого підграфа LangGraph для кожного підагента. Це забезпечує справжню паралельність виконання. Кожен підагент виконує ідентичний, але незалежний цикл: формулює оптимізовані пошукові запити, виконує Пошук через Tavily API, отримує структуровані результати (посилання, снипети, метадані) і передає їх на етап Стиснення. Стиснення є критично важливим: замість передачі повних текстів знайдених джерел, підагент, використовуючи LLM, аналізує, синтезує та компресує інформацію у короткий, змістовий виклад по своїй підтемі, обов'язково додаючи посилання на першоджерела. Цей підхід ефективно боротися з проблемою "інфляції токенів" та запобігає переплутуванню контекстів між різними агентами.

– Підграф Synthesis (Синтез: агрегація та генерація звіту). Після завершення роботи всіх підагентів їх стислі висновки, разом з оригінальним брифом, потрапляють до вузла Агрегації. Цей вузол готує консолідований набір даних для фінальної генерації. Далі, LLM-вузол генерації (який може бути окремою, більш потужною моделлю) отримує ці дані та створює Когерентний фінальний звіт. Завданням цього вузла є не механічне об'єднання текстів, а створення нового, цілісного документа з послідовною аргументацією, усуненням суперечностей, дотриманням заданої структури та інтеграцією джерел у вигляді цитат або загального списку літератури. Згенерований звіт записується назад у спільний стан, сигналізуючи про завершення робочого циклу.

Для забезпечення діагностики, налагодження та контролю якости, архітектура ODR інтегрована з платформою LangSmith. Цей інструмент забезпечує всебічну інтроспекцію системи в реальному часі. Він підключається до графа LangGraph, фіксуючи кожен крок виконання: активацію вузлів, передачу стану, виклики зовнішніх інструментів (Tavily) та промпти LLM. Крім того, LangSmith надає повну візуалізацію еволюції стану, дозволяючи досліднику відстежити, як саме формувався бриф, які підзавдання були створені, які пошукові запити генерувалися та як стискалися результати. Ця спостережуваність є незамінною для

академічного використання, оскільки забезпечує прозорість процесу прийняття рішень агентом та дозволяє валідувати якість і релевантність кожного проміжного кроку.

Таким чином, деталізована архітектура ODR демонструє високий ступінь модульності, детермінованості та контролю. Взаємодія компонентів, опосередкована централізованим станом та оркестрована гнучким графом, забезпечує ефективну реалізацію складного багатоагентного дослідження. Інтеграція з інструментами моніторингу робить цей процес повністю спостережуваним, що відповідає вимогам наукової строгості та дозволяє адаптувати систему під специфічні завдання в галузі телекомунікацій.

3.2 Адаптація системи під предметну область

У Архітектура ODR є універсальним інструментом, розробленим для автоматизації наукового дослідження в різних галузях. Її сила полягає саме в гнучкості та модульності: ядро системи у вигляді графа LangGraph реалізує загальну логіку workflow (Scoping, Research, Synthesis), абстрагуючись від конкретного наповнення. Однак для досягнення максимальної ефективності та релевантності результатів при застосуванні до вузькопрофільних дисциплін, таких як телекомунікації, необхідно налаштувати системні компоненти, що безпосередньо взаємодіють з моделлю штучного інтелекту та зовнішніми джерелами. Таким чином, адаптація не передбачала глибоких змін в архітектурі рушія, а зосередилася на кастомізації його "інтелектуального наповнення" — стратегій планування, пошуку та синтезу, закодованих у промптах і конфігурації агентів.

Метою адаптації було трансформувати універсальну дослідницьку платформу в спеціалізованого асистента, зданого "мислити" категоріями телекомунікаційної інженерії, орієнтуватися на авторитетні галузеві джерела та формулювати вичерпні, технічно точні висновки. Це дозволяє зберегти всі переваги стабільної та відлагодженої архітектури ODR, одночасно націливши її на вирішення специфічних завдань, таких як аналіз нових протоколів зв'язку, порівняння архітектурних рішень чи моніторинг технологічних трендів.

Адаптація була реалізована шляхом створення двох ключових модулів, які інтегруються в базову систему як заміщуючі компоненти:

Модуль `telecom_researcher.py` (Граф доменно-орієнтованого дослідження). Цей файл містить визначення спеціалізованого графа LangGraph, що успадковує основну структуру та логіку виконання з оригінального `deep_researcher`, але модифікований для роботи з телекомунікаційним контекстом. Його основна відмінність полягає у явному підключенні нового набору промптів та, потенційно, у точній настройці параметрів виконання (наприклад, глибини ітерацій пошуку для технічних тем). Сам граф як механізм оркестрації агентів залишається незмінним; зміни стосуються виключно "начинки", якою живляться агенти на кожному вузлі.

Модуль `teleprompts.py` (Бібліотека доменних промптів). Цей модуль є серцем адаптації. Він містить набір структурованих текстових інструкцій (промптів), оптимізованих для кожного етапу роботи системи та для кожної ролі агента (Супервайзер, Підагент, Генератор звіту) у контексті телекомунікацій. Оригінальні промпти ODR є загальними та універсальними. Завданням `teleprompts.py` було перевизначити їх, зосередившись на:

- Направленості на авторитетні джерела: Інструкції явно звертають увагу агентів на необхідність пошуку інформації в офіційних стандартах (3GPP, ITU-T, IEEE, ETSI), технічній документації провідних вендорів, рецензованих наукових журналах та матеріалах ключових конференцій (наприклад, GLOBECOM, ICC).
- Акцент на повноту та технічну глибину: Промпти наказують агентам уникати поверхневих відповідей та надмірного стиснення (сумаризації) на ранніх етапах. Замість цього вони мають витягувати та структурувати конкретні технічні параметри, архітектурні принципи, порівняльні переваги та недоліки, цифрові показники ефективності.
- Мову та категорії предметної області: Інструкції використовують професійну термінологію, що дозволяє LLM точніше інтерпретувати запити та формулювати результати мовою цільової галузі.

Адаптація зводиться до заміни "інтелектуальної прошивки" системи — набору текстових інструкцій для LLM — при абсолютно недоторканому "апаратному забезпеченні" у вигляді графа виконання та механізмів оркестрації. Цей підхід підтверджує високу гнучкість ODR та дозволяє ефективно використовувати його для автоматизації наукового аналізу саме в предметній області телекомунікацій, забезпечуючи релевантність, глибину та технічну точність отриманих результатів.

3.3 Інтеграція та запуск системи

3.3.1 Локальне розгортання

Для експериментальної роботи та подальшого використання в дослідницькому процесі необхідним кроком є локальне розгортання системи ODR. Це дозволяє отримати повний контроль над середовищем виконання, здійснювати налагодження, модифікацію компонентів та інтегрувати власні інструменти. Процес розгортання є стандартизованим та добре документованим розробниками, що забезпечує його відтворюваність на різних платформах. Хоча для роботи системи не потрібна специфічна інтегрована середовища розробки (IDE), в межах даного дослідження для зручності редагування коду, налагодження та управління віртуальним середовищем було використано PyCharm.

Першим етапом є отримання актуальної версії вихідного коду з офіційного репозиторію. Це забезпечує доступ до всіх модулів системи, включаючи базову архітектуру та інструменти для запуску:

```
>git clone https://github.com/langchain-ai/open_deep_research.git
>cd open_deep_research
```

Для ізоляції залежностей проекту та уникнення конфліктів з глобальним середовищем Python обов'язковим є створення та активація віртуального середовища. Система підтримує використання сучасного менеджера пакетів `uv` для ефективного керування середовищем:

```
>uv venv
source .venv/bin/activate  # На платформах Windows: .venv\Scripts\activate
```

Після активації віртуального середовища необхідно встановити всі Python-

пакети, необхідні для функціонування системи. Залежності чітко визначені у файлі `pyproject.toml`. Команда `uv sync` автоматично аналізує цей файл, створює блокування залежностей (lock file) та встановлює всі необхідні пакети, включаючи фреймворки LangChain і LangGraph, клієнти API та інші службові бібліотеки:

```
>uv sync
# Альтернативний варіант:
# >uv pip install -r pyproject.toml
```

Система керується через змінні середовища, що надає гнучкість у виборі моделей LLM, пошукових бекендів (таких як Tavily) та інших параметрів без зміни коду [7]. Для початку роботи необхідно ініціалізувати файл конфігурації `.env` на основі наданого шаблону:

```
>cp .env.example .env
```

ODR використовує архітектуру, де серверна частина (агентний граф) працює як локальний сервер, а керування відбувається через веб-інтерфейс LangGraph Studio. Для запуску всього стека застосунку використовується команда `langgraph dev`:

```
>uvx --refresh --from "langgraph-cli[inmem]" --with-editable . --python 3.11 langgraph dev --allow-blocking
```

Ця команда виконує кілька дій:

1) Використовує `uvx` для одноразового виконання пакета `langgraph-cli`.
2) Запускає сервер розробки (`dev`) з графом, визначеним у поточному каталозі (`--with-editable .`).
3) Вказує версію Python (`--python 3.11`).
4) Флаг `--allow-blocking` дозволяє агентам виконувати довготривалі синхронні операції (наприклад, пошук в інтернеті).

Після успішного запуску в терміналі з'явиться інформація про доступні ендпоінти:

– API: `http://127.0.0.1:2024` – JSON API сервер для програмної взаємодії.
– Studio UI: `https://smith.langchain.com/studio/?baseUrl=http://127.0.0.1:2024` –

URL веб-інтерфейсу для візуального керування.

– API Docs: `http://127.0.0.1:2024/docs` – інтерактивна документація API (Swagger/OpenAPI).

Відкривши посилання на Studio UI в браузері, користувач отримує доступ до інтерфейсу, де можна:

– Ввести дослідницький запит у текстове поле (наприклад, "Проаналізуй останні тенденції у розвитку мереж 6G").
– Налаштувати параметри агента у вкладці "Manage Assistants" (вибір моделі, глибини пошуку тощо).
– Запустити виконання, натиснувши "Submit".
– Спостерігати за ходом виконання графа в реальному часі, переглядати проміжні результати пошуку, логи агентів та фінальну відповідь.

Процес локального розгортання є чітким та стандартизованим, що дозволяє досліднику швидко отримати працездатне середовище для проведення експериментів з автоматизації наукового аналізу в галузі телекомунікацій. Використання віртуального середовища та конфігураційних файлів забезпечує ізоляцію та відтворюваність експериментів, що є важливою вимогою в академічній роботі.

3.3.2 Налаштування доступу до LLM через OpenRouter

Незважаючи на успішне локальне розгортання системи ODR, її функціональність залишається обмеженою без налаштованого доступу до потужних LLM. Архітектура системи побудована навколо інтелектуальних агентів, які для виконання ключових операцій — аналізу запитів, планування дослідження, синтезу текстів — потребують інтеграції з зовнішніми LLM API. Відкриті (open-source) моделі, що можуть бути розгорнуті локально, часто не мають достатньої контекстної пам'яті або якості відповідей для складних багатоетапних досліджень, тоді як найбільш потужні та спеціалізовані моделі, такі як GPT-4, Claude 3 або Gemini Pro, надаються комерційними провайдерами на платній основі. Без

налаштування відповідних ключів доступу (API keys) система не зможе виконувати своє основне призначення. Найпростішим шляхом є використання прямого API від OpenAI, вказавши власний ключ у змінній середовища OPENAI_API_KEY. Однак такий підхід має обмеження у виборі моделей та може виявитись економічно неефективним при інтенсивному використанні. Пряме використання API окремих провайдерів (OpenAI, Anthropic, Google) потребує окремих облікових записів, ключів та інтеграцій, що ускладнює управління та підвищує вартість експериментів через фіксовані місячні підписки.

Для подолання цих обмежень у даній роботі було обрано сервіс агрегації OpenRouter, що виступає уніфікованим API-шлюзом до сотень моделей від десятків провідних провайдерів. Його бізнес-модель базується на кредитній системі, коли користувач купує пакет кредитів, які потім витрачаються пропорційно використанню будь-якої з доступних моделей. Це надає ключові переваги для академічного дослідження: значно нижчі витрати за рахунок відсутності обов'язкових щомісячних платежів, можливість миттєвого порівняння різних моделей на одному завданні та спрощена адміністрація через єдиний API-ключ.

Таблиця 3.1

Порівняльні характеристики доступу

Критерій	**Прямий доступ до провайдера**	**Доступ через OpenRouter**
Економічна модель	Фіксована щомісячна підписка або pay-per-use.	Pay-per-use з попередньою купівлею кредитів; немає обов'язкових щомісячних платежів.
Управління ключами	Окремий ключ для кожного провайдера.	Один API-ключ для всіх моделей.
Вибір моделей	Обмежений моделями одного провайдера.	Єдиний інтерфейс для сотень моделей від десятків провайдерів.
Складність інтеграції	Потрібно адаптувати код під кожен SDK.	Повна сумісність з OpenAI API, що мінімізує зміни коду.

Технічною основою інтеграції є те, що OpenRouter надає повністю сумісне з OpenAI API, що дозволяє використовувати стандартні клієнтські бібліотеки. Щоб

перенаправити запити до OpenRouter, достатньо змінити два параметри при ініціалізації об'єкта:

1) `base_url` (або `openai_api_base`): встановити на `"https://openrouter.ai/api/v1"`.
2) `api_key`: вказати ключ, отриманий в обліковому записі OpenRouter (`OPENROUTER_API_KEY`).

Для гнучкого використання як прямих API провайдерів, так і OpenRouter, було розширено центральну Pydantic-модель конфігурації у файлі `configuration.py`. Було додано обов'язкове поле для визначення базової URL-адреси API, за замовчуванням націлене на OpenRouter:

```
openai_api_base: str = Field(
    default="https://openrouter.ai/api/v1",
    metadata={
        "x_oap_ui_config": {
            "type": "text",
            "default": "https://api.openai.com/v1",
            "description": "Base URL for the LLM API endpoint",
        }
    }
)
```

Крім того, для кожного типу моделі, що використовується в різних фазах (дослідження, стиснення, генерація звіту), було введено окремі поля для вказівки провайдера (наприклад, `"openai"`, `"anthropic"`). Це критично важливо, оскільки внутрішній метод `init_chat_model` системи потребує чіткого вказівника для коректної ініціалізації моделей, які не належать до екосистеми OpenAI. Приклад для моделі синтезу:

```
summarization_model_provider: str = Field(
    default="openai",
    metadata={
        "x_oap_ui_config": {
            "type": "text",
            "default": "openai",
            "description": "Provider name for the summarization model (e.g.,
'openai', 'anthropic')",
        }
    }
)
```

Незважаючи на те, що значення за замовчуванням встановлено як "openai", це поле виконує технічну роль ідентифікатора для внутрішньої логіки ініціалізації. На практиці, при використанні OpenRouter, значення цього поля може залишатися

"openai", навіть якщо фактично викликається модель Anthropic Claude. Це є тимчасовим рішенням оскільки основним критерієм вибору моделі в даній архітектурі є її ім'я (наприклад, "anthropic/claude-3-opus"), що передається разом із запитом на вказаний openai_api_base. Аналогічні поля було додано для research_model_provider, compression_model_provider та final_report_model_provider.

Наступним кроком була модифікація фабрики моделей для підтримки нових конфігураційних полів. У головному файлі графа, де відбувається ініціалізація конфігурованих моделей, було розширено список конфігурованих полів (configurable_fields). Це дозволяє передавати нові параметри (openai_api_base та model_provider) з конфігурації безпосередньо в функцію створення екземпляра моделі. У головному графі `telecom_researcher.py` сигнатура функції `init_chat_model` була оновлена, щоб приймати додаткові поля:

```
configurable_model = init_chat_model(
    configurable_fields=("model", "max_tokens", "api_key",
                         "openai_api_base", "model_provider",)
)
```

Після внесення описанних змін процес налаштування для кінцевого дослідника стає максимально простим.

– У файлі .env встановлює змінну OPENAI_API_KEY рівною отриманому ключу OpenRouter. Система сприйматиме його як ключ для доступу до агрегованого API;
– Поле "Provider" може залишатися значенням за замовчуванням ("openai"), оскільки фактичний провайдер визначається через ім'я моделі в першому кроці;
– Поле `research_model` тепер приймає ідентифікатори моделі у форматі OpenRouter, такі як `openai/gpt-4o`, `anthropic/claude-3.5-sonnet` або `meta-llama/llama-3.1-70b-instruct`.

Цей підхід надає досліднику гнучкість: можна швидко перемикатися між різними потужними моделями для тестування або вибрати оптимальну за співвідношенням "вартість-якість" для конкретного етапу роботи, використовуючи лише один ключ доступу.

3.4 Демонстрація роботи системи

3.4.1 Приклад запиту та аналіз роботи агента

Для практичної валідації можливостей адаптованої системи та ілюстрації її робочого циклу було проведено демонстраційний запуск з типовим для дослідження телекомунікацій запитом. Важливо зазначити специфіку термінології, вживаної в екосистемі LangChain: граф (graph) визначає програмну логіку та алгоритм дій агента — послідовність і умови виконання вузлів, тоді як асистент (assistant) — це конкретний набір конфігураційних параметрів для екземпляра цього графа, таких як вибір моделей LLM, розмір контекстного вікна, правила використання інструментів тощо.

Робота розпочалася з запуску серверної частини системи. У терміналі, активованому в кореневій директорії проекту, було виконано команду:

```
uvx --refresh --from "langgraph-cli[inmem]" --with-editable . --python 3.11 langgraph dev --allow-blocking
```

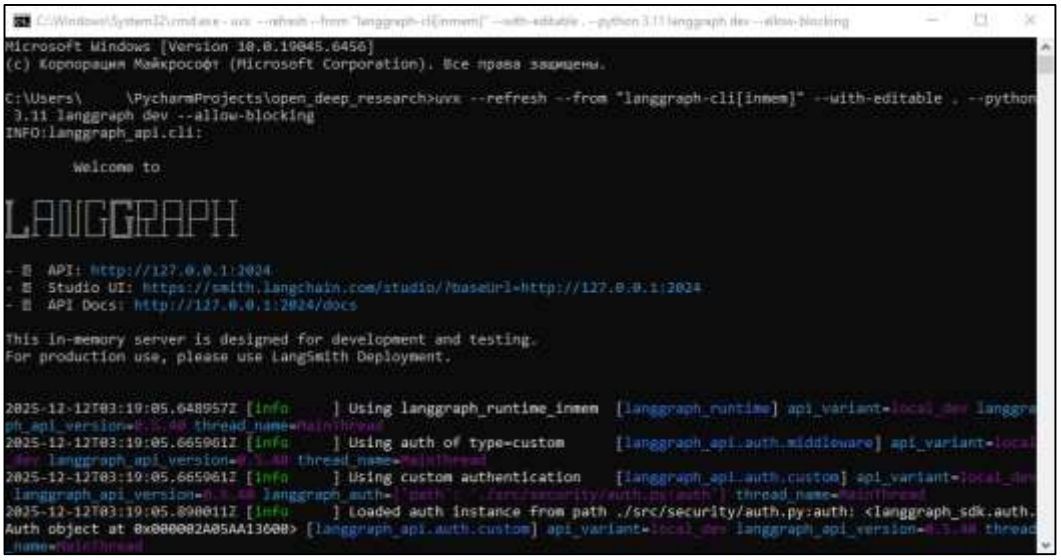

Рис.3.3. Запуск локального серверу системи з терміналу

Ця команда ініціювала локальний сервер LangGraph, який обслуговує API для взаємодії з агентом, та автоматично відкрила у системному браузері веб-інтерфейс LangSmith Studio.

Критично важливим аспектом є те, що весь агент (граф) виконується локально на машині дослідника. LangSmith Studio в даному контексті виступає виключно як зручний веб-інтерфейс (фронтенд) для візуалізації, конфігурації та керування. Уся обробка запитів, виклики моделей та робота з інструментами відбуваються в локальному середовищі Python.

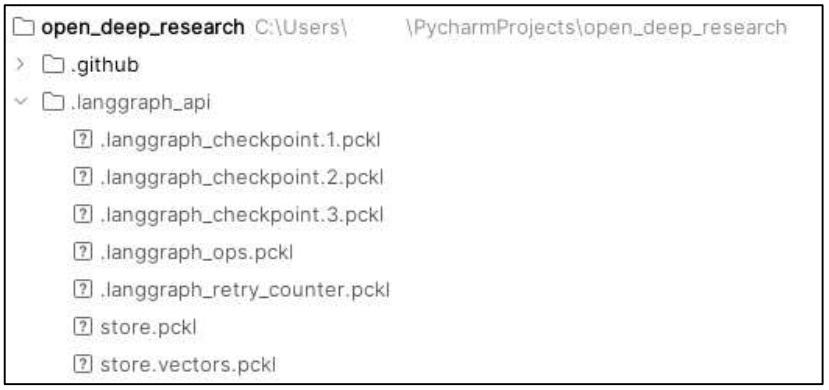

Рис.3.4. Локальне сховище даних проекту

Всі проміжні дані, логи виконання та конфігурації зберігаються у локальній папці `.langgraph_api`, а не на віддалених серверах LangChain. Це забезпечує повну конфіденційність даних та автономність роботи. Хоча вхід в обліковий запис LangSmith надає доступ до додаткових хмарних функцій (спільна робота, розширена аналітика), для базового використання власних агентів він не є обов'язковим.

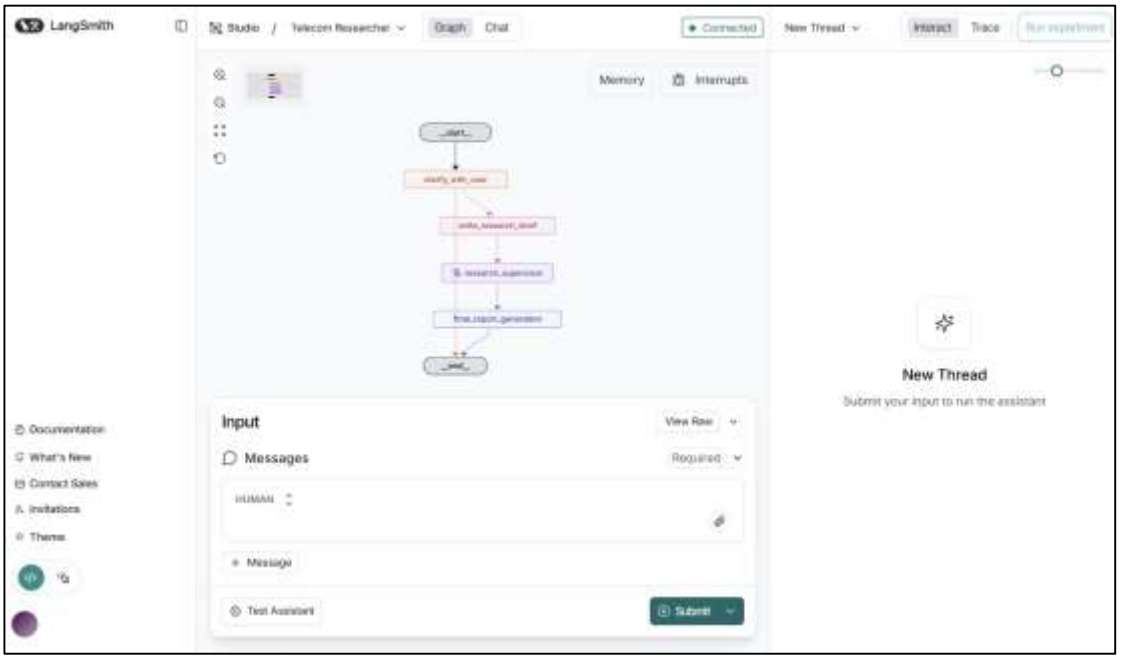

Рис.3.5. Головний інтерфейс системи в середовищі LangSmith

Інтерфейс представляє собою інтегроване середовище для розробки та виконання багатоетапних агентів [17]. Його структуру можна розділити на кілька ключових зон:

- Центральна область: Візуалізує блок-схему (граф) агента, де кожен вузол відповідає певному етапу логіки (Scoping, Research, Synthesis), а ребра відображають умовні переходи. Це надає чітке уявлення про програму потоку виконання;
- Ліва бічна панель: Забезпечує навігацію по ширшим функціям платформи LangSmith (документація, управління командою, налаштування проєкту);
- Верхня панель вкладок: Дозволяє перемикатися між переглядом графів (агентів), інтерфейсом чату для взаємодії та панеллю трасування (tracing) для детального аналізу окремих виконань;
- Права область (Потоки/Чати): Основна робоча зона для тестування. Тут створюються нові потоки (threads) — сесії взаємодії з агентом. У цій області відображається хід виконання графа в реальному часі, проміжні результати, а також фінальна відповідь;
- Нижня панель: Містить поле для введення повідомлень користувача, кнопку відправки (`Submit`) та доступ до конфігурації поточного асистента (`Manage Assistants`).
- У вкладці "Manage Assistants" було створено новий асистент на основі кастомного графа `telecom_researcher`. У конфігурації було налаштовано:
- Базовий URL API (`openai_api_base`): `https://openrouter.ai/api/v1`;
- Назви моделей: Для етапів дослідження та генерації звіту було обрано моделі, доступні через OpenRouter (наприклад, `openai/gpt-5.1` для аналізу);
- Розміри контекстних вікон (`max_tokens`): Стандартні значення було збільшено для обробки великих обсягів технічної інформації, що характерно для телекомунікаційної предметної області.

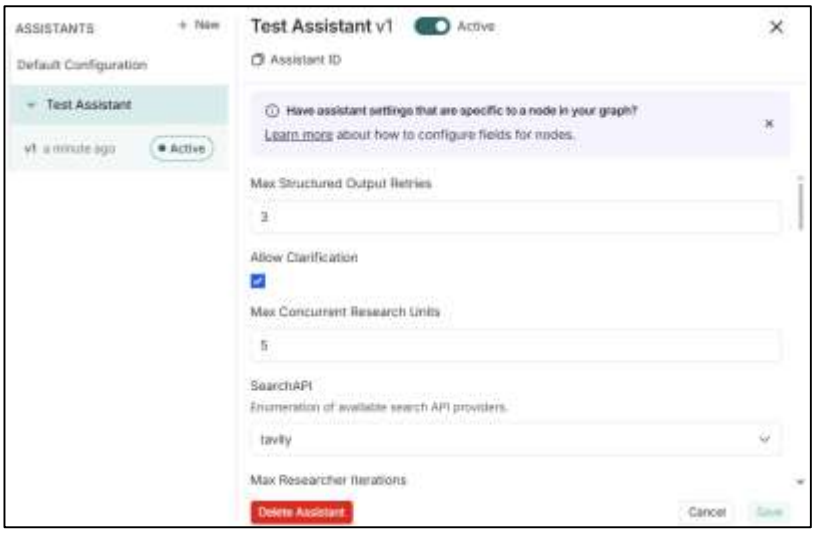

Рис.3.6. Інтерфейс створення налаштувань для агента

Після збереження конфігурації асистент стає доступним для використання. У новому потоці (thread) у поле введення повідомлення було направлено тестове запитання, що моделює реальну дослідницьку потребу:

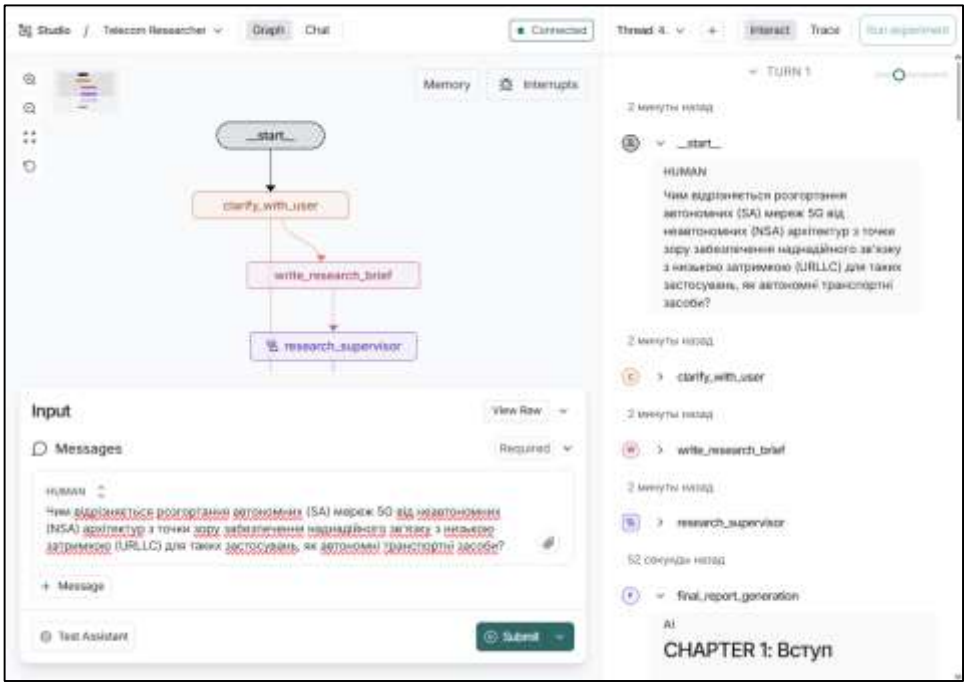

Рис.3.7. Заповнення запиту та візуалізація процесу логіки агента

Після натискання кнопки `Submit` система розпочала виконання. У правій панелі в режимі реального часу став відображатися хід виконання графа:

1) Фаза Scoping: Система, використовуючи LLM, проаналізувала запит, ідентифікувала ключові терміни (SA, NSA, URLLC) та сформулювала стислий бриф для подальшого дослідження;

2) Фаза Research: Агент-супервайзер розбив тему на підзадачі (наприклад: "архітектурні відмінності SA vs NSA", "вимоги URLLC", "вимоги для автономних транспортних засобів"). Для кожної підзадачі створювався окремий підагент, який паралельно виконував пошук через Tavily API, аналізував технічні статті, стандарти 3GPP та звіти вендорів, після чого стискав знайдену інформацію у структурований виклад з посиланнями;
3) Фаза Synthesis: Усі стислі висновки підагентів були агреговані та передані LLM для генерації фінального звіту. В результаті було отримано когерентну, всебічну відповідь, що структуровано порівнювала архітектури SA та NSA в контексті забезпечення низької затримки та надійності, посилаючись на актуальні технічні джерела.

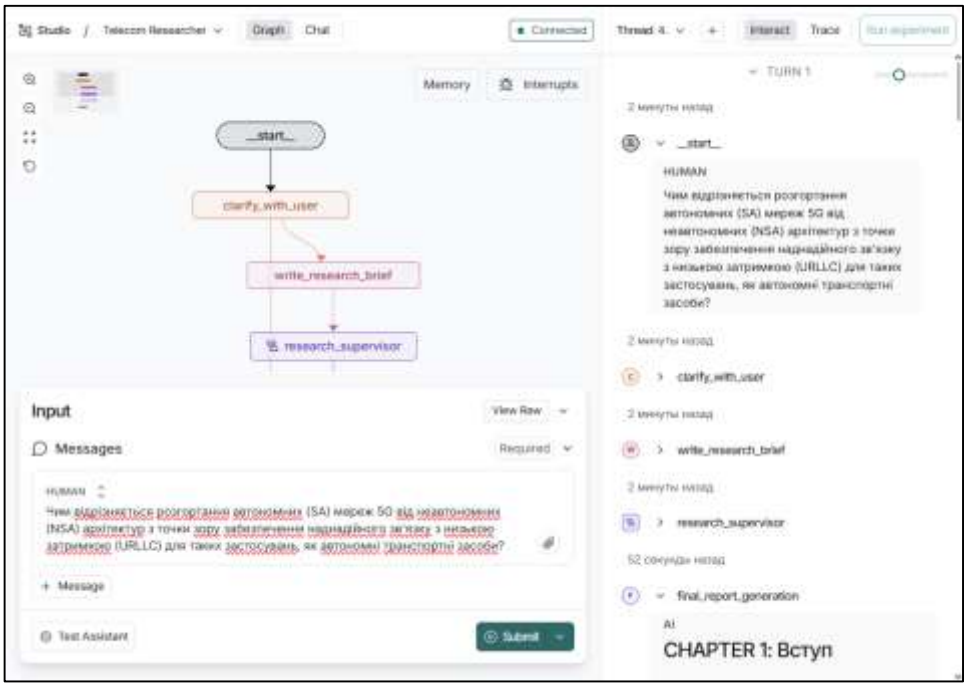

Рис.3.8. Заповнення запиту та візуалізація процесу логіки агента

Таким чином, демонстрація наочно підтвердила практичну працездатність адаптованої системи, її здатність автономно планувати та виконувати комплексне наукове дослідження в заданій предметній області, трансформуючи складний технічний запит у структурований, джерелами підкріплений аналітичний звіт.

3.4.2 Оцінка результатів та обговорення обмежень

Отриманий у результаті виконання демонстраційного запиту документ був підданий критичному аналізу з метою оцінки якості роботи адаптованої системи. Система продемонструвала видатні результати за критеріями точності, повноти, логічної структури та практичної цінності. Отриманий за 8 хвилин 20-сторінковий технічний звіт більшою частиною відповідає стандартам академічного огляду та інженерного аналізу в галузі телекомунікацій. Проте, для комплексної оцінки необхідно розглянути не лише сильні сторони, але й економічні аспекти, порівняльні характеристики та існуючі обмеження.

Ключовим практичним результатом є екстремально низька собівартість отриманого аналізу. Витрати на генерацію комплексного 20-сторінкового документа з використанням потужних моделей через OpenRouter (у даному випадку — `gpt-5.1`) склали приблизно 0.4 USD. Ця сума враховує витрати токенів на всі етапи: уточнення запиту, паралельний пошук та аналіз декількома підагентами, синтез фінального звіту. Така вартість робить систематичне використання подібного інструменту економічно доцільним навіть для індивідуальних дослідників або невеликих лабораторій.

Цю ефективність особливо яскраво видно при порівнянні з двома альтернативами:

1) Ручна робота експерта: Для підготовки аналогічного за глибиною та обсягом огляду кваліфікованому фахівцеві знадобилося б від кількох днів до кількох тижнів роботи, що суттєво перевищує як часові, так і фінансові витрати.
2) Пряме використання загальнодоступних чат-інтерфейсів (наприклад, безкоштовний ChatGPT): Такі інструменти мають серйозні обмеження, що роблять подібне дослідження неможливим. По-перше, вони мають жорстко обмежений контекстне вікно, недостатнє для обробки та синтезу десятків технічних джерел. По-друге, їхні відповіді зазвичай носять загальний, нетехнічний характер, без глибокого занурення в специфіку стандартів. По-третє, вони не мають інтегрованого механізму пошуку в реальному часі та автоматичного цитування джерел, що є критичним для наукової праці.

Незважаючи на вражаючі результати, система має ряд обмежень, обумовлених як її архітектурою, так і загальними межами сучасних LLM.

- Якість та глибина цитування: Хоч система надає численні посилання на авторитетні джерела, цитування часто є загальним. Наприклад, вказується номер специфікації 3GPP TS 23.501, але без посилання на конкретний розділ, сторінку або версію документа. Це ускладнює пряму верифікацію фактів і змушує дослідника проводити додаткову ручну роботу з пошуку в першоджерелі. Реалізація механізму точного цитування з прив'язкою до фрагментів текстів є окремою складною науково-інженерною задачею, що виходить за рамки даної роботи;
- Залежність від якості промптів: Результат безпосередньо залежить від якості промптів у модулі `teleprompts.py`. Незважаючи на адаптацію під телекомунікації, промпти можуть не охопити всі нюанси конкретного підзапиту. Наприклад, система може недостатньо критично оцінити обмеження чи конфліктуючі дані з різних джерел. Дальша робота над удосконаленням промптів, включення в них інструкцій щодо критичного аналізу та виявлення суперечностей може суттєво підвищити якість висновків;
- Актуальність інформації: Система працює з найновішими джерелами через Tavily, однак існує природнє відставання між публікацією нових стандартів чи досліджень та їх індексацією пошуковими системами. Для критично важливих рішень необхідно проводити додаткову перевірку актуальності знайдених джерел;
- Відсутність глибокого критичного та прогностичного аналізу: Система ефективно синтезує наявну інформацію, але її здатність до справжньо критичної оцінки технологій, виявлення прихованих ризиків або формування власних прогностичних сценаріїв обмежена. Це залишається за людиною-експертом.

Проведена демонстрація та наступний аналіз підтверджують, що адаптована система ODR є високоефективним інструментом для автоматизації первинного та

вторинного наукового аналізу в галузі телекомунікацій. Вона дозволяє в режимі, близькому до реального часу, отримувати комплексні, добре структуровані та технічно точні огляди з мінімальними фінансовими витратами.

Система не замінює повністю дослідника, але виступає потужним синергетичним інструментом, що бере на себе рутинну, але трудомістку роботу зі збору, первинної обробки та структурування великих масивів інформації. Це дозволяє людському експерту зосередитись на вищорівневих завданнях: критичній оцінці, формуванні гіпотез, дизайні експериментів та інтерпретації результатів. Подальша робота має бути спрямована на вдосконалення механізмів цитування, розвиток критичного мислення агентів та інтеграцію системи в повноцінний цикл наукового дослідження.

Висновки

Розробка та налаштування ШІ-асистента на основі фреймворку Open Deep Research для автоматизації наукового аналізу в галузі телекомунікацій продемонструвала перехід від теоретичних архітектурних моделей до практичної, працездатної системи, здатної автономно виконувати складні дослідницькі workflow. Вибір ODR як базової платформи підтвердив його придатність як універсального та гнучкого інструменту, що поєднує потужність сучасних LLM з чітко структурованою логікою виконання, реалізованою через симбіоз LangChain та LangGraph. Ця архітектура, заснована на концепції станомісткого орієнтованого графа, дозволила втілити циклічну парадигму «Мислення-Дії-Спостереження» у формі адаптивного трифазного процесу, що імітує ключові етапи наукового мислення: від уточнення проблеми та формування брифу через планування, паралельний пошук і аналіз до синтезу когерентного звіту.

Критичною відмінністю реалізованої системи від традиційних RAG-підходів стала її фундаментальна орієнтація на пошук інформації в реальному часі через спеціалізовані API, такі як Tavily Search. Це дозволило подолати обмеження, пов’язані зі статичністю попередньо індексованих корпусів, та забезпечити доступ до найновіших наукових публікацій, технічних стандартів, ринкових звітів і інформації з профільних ресурсів, що є вирішальним для динамічної галузі

телекомунікацій. Інтеграція централізованого об’єкта стану (State Object) та модульної системи конфігурації забезпечила прозорість потоку даних, детермінованість виконання та спостережуваність на всіх етапах, що є ключовим для наукової відтворюваності та налагодження.

Адаптація універсальної платформи під специфіку телекомунікаційної предметної області була елегантно реалізована не через зміну ядра системи, а шляхом кастомізації її «інтелектуального наповнення». Створення спеціалізованих модулів `telecom_researcher.py` та `teleprompts.py` дозволило переналаштувати стратегії планування, пошуку та синтезу, спрямувавши агентів на використання професійної термінології, авторитетних галузевих джерел (на кшталт 3GPP, ITU, IEEE) та акцентуючи увагу на технічній глибині аналізу. Цей підхід підкреслює силу модульності ODR, що дозволяє трансформувати універсальний інструмент у високоспеціалізованого асистента з мінімальними змінами в архітектурі.

Практична демонстрація роботи системи на прикладі комплексного запиту щодо архітектур 5G SA та NSA та вимог URLLC наочно підтвердила її працездатність та ефективність. Система продемонструвала здатність автономно декомпозувати завдання, планувати паралельне виконання підагентами, здійснювати ітеративний пошук і аналіз, а потім синтезувати об’ємний, структурований технічний звіт із посиланнями на джерела. При цьому економічна ефективність виявилася вражаючою – собівартість генерації всебічного 20-сторінкового аналізу склала лише частки долара завдяки використанню агрегованого API OpenRouter, що робить подібний інструмент доступним для індивідуальних дослідників та академічних установ.

Однак експериментальне впровадження також виявило певні системні обмеження, які окреслюють напрями для подальшого вдосконалення. Серед них – недостатня деталізація цитування (відсутність прив’язки до конкретних розділів стандартів), залежність якості висновків від точності та повноти промптів, а також обмежена здатність системи до глибокого критичного та прогностичного аналізу, що залишається прерогативою людини-експерта.

Розроблений ШІ-асистент не замінює дослідника, а виступає потужним

синергетичним інструментом, що бере на себе рутинну, інформаційно-емоційну роботу зі збору, первинної обробки та структурування знань. Це дозволяє фахівцю зосередитися на вищорівневих інтелектуальних завданнях: формуванні гіпотез, критичній оцінці, інтерпретації результатів та стратегічному плануванні подальших досліджень. Подальша розробка має бути спрямована на вдосконалення механізмів точного цитування, розвиток критичного мислення агентів та їх глибшу інтеграцію в повноцінний цикл наукової діяльності, що відкриває нові горизонти для прискорення генерації знань у складних технічних галузях, таких як телекомунікації.

РОЗДІЛ 4

СТАРТАП-ПРОЄКТ

4.1 Концептуальні засади стартап-проєкту

Науково-дослідні інновації досягають повної цінності лише тоді, коли вони переходять від лабораторного прототипу до практичного інструменту, здатного вирішувати реальні проблеми у промисловості та науці. Даний розділ присвячений трансформації спеціалізованого ШІ-асистента для телекомунікацій, розробленого в попередніх розділах, у життєздатну бізнес-модель. Основна мета — продемонструвати, як глибоко адаптоване під галузь програмне рішення, побудоване на відкритих технологіях, може стати основою для інноваційного стартапу, що закриває критичну потребу ринку в автоматизації складного технічного аналізу.

Ключовою концепцією проєкту є переосмислення ролі open-source інструменту. Подібно до того, як компанія Red Hat збудувала глобальний бізнес навколо підтримки, вдосконалення та інтеграції операційної системи Linux для корпоративного сегменту, запропонований стартап будує свою цінність не на власній унікальній архітектурі, а на експертній, глибокій адаптації та налаштуванні платформи Open Deep Research під специфічні потреби телекомунікаційної галузі. Це включає:

- Галузеву спеціалізацію логіки: Налаштування промптів, стратегій пошуку та аналізу для роботи з професійною термінологією, авторитетними джерелами, та технічними стандартами;
- Інтеграцію галузевих інструментів: Підключення до спеціалізованих пошукових API та баз знань, критичних для телекомунікацій;
- Формування готового результату: Трансформацію сили алгоритму в конкретну послугу — генерацію всебічних, структурованих технічних звітів з актуальними посиланнями на джерела.

Наукова розробка довела практичну працездатність системи, її здатність автономно виконувати комплексний науковий пошук та аналіз, генеруючи за

декілька хвилин технічний звіт, який за обсягом і структурою відповідає тижням ручної роботи експерта. Експериментальні дані показали екстремально високу економічну ефективність: вартість створення 20-сторінкового аналітичного документа склала близько 0.4 USD. Цей факт є фундаментальним для бізнес-моделі, оскільки перетворює науковий результат (алгоритм) на конкурентну перевагу (вартість послуги, що на порядки нижча за альтернативи).

Стартап-проєкт, отже, позиціонується не як «винахід нового ШІ», а як «Telecom Research Assistant as a Service» — сервіс, що надає телекомунікаційним компаніям, дослідницьким інститутам та аналітикам доступ до потужного, спеціалізованого інструменту через зручну модель підписки або разових замовлень. Це дозволяє зосередитись на головному — унікальній цінності, що полягає в глибокому знанні предметної області та оптимізації процесу роботи з інформацією, а не на розробці базової інфраструктури з нуля. Таке формування продукту відповідає сучасній тенденції, коли ринок потребує не ще одного загального інструменту, а високоспеціалізованих рішень, що посилюють експертність у складних технічних галузях.

4.2 Опис продукту

Ядром стартап-проєкту є продукт — сервіс, який трансформує розроблений у попередніх розділах спеціалізований ШІ-асистент із інструменту дослідження у комерційно доступну послугу. Продукт отримує назву «Telecom Research Assistant as a Service» (TRaaS), що точно відображає його суть: надання автоматизованих, всебічних технічних аналітичних звітів у галузі телекомунікацій через модель сервісу.

Суть продукту полягає в переході від програмного забезпечення до результату. Клієнт не купує доступ до коду чи налаштовує власну інфраструктуру, а отримує готовий, структурований документ — технічний огляд, конкурентний аналіз, звіт про стан технології чи патентний огляд, сформований автономною системою на основі його запиту. Ключовою відмінністю від загальних ШІ-інструментів є глибока галузева предметна підготовка системи, яка вже інкорпорує знання про авторитетні джерела (3GPP, IEEE Xplore, arXiv, ITU-R), специфічну

термінологію та критичні аспекти аналізу телекомунікаційних технологій.

Ядро цінності продукту формується чотирма взаємопов'язаними стовпами:

1. Економія часу висококваліфікованих експертів. Система скорочує цикл первинного та вторинного наукового аналізу з тижнів та місяців ручної роботи до годин або навіть хвилин автоматизованого процесу. Як продемонстровано в практичній валідації, комплексний 20-сторінковий технічний звіт може бути сформовано за 8-10 хвилин роботи системи. Це звільняє інженерів та науковців від рутинного збору та систематизації інформації, дозволяючи зосередитись на інтерпретації, критичній оцінці та стратегічному прийнятті рішень.

2. Безпрецедентна економічна ефективність. Собівартість послуги є ключовою перевагою. Генерація всебічного аналітичного звіту через оптимізоване використання LLM-моделей та API пошуку становить приблизно 0.4 USD (за даними демонстраційного запуску). Ця сума на порядки нижча за вартість робочого дня галузевого аналітика або консультанта, роблячи систематичний технічний аналіз доступним не лише для корпорацій, а й для малих дослідницьких груп, стартапів та академічних установ.

3. Галузева експертиза «в коробці»:

– цільовий пошук: Система «знає», де шукати інформацію для телекомунікацій, віддаючи пріоритет науковим репозиторіям, стандартизаційним органам та технічним блогам вендорів;
– розуміння контексту: Завдяки спеціалізованим промптам система коректно інтерпретує абревіатури (напр., NSA, SA, mMTC), технічні концепції (напр., network slicing, beamforming) та розрізняє схожі терміни в різних контекстах;
– структура відповідна галузі: Форматування та логіка звітів адаптовані під технічну документацію, включаючи розділи з вимогами, архітектурними порівняннями, викликами та посиланнями на першоджерела.

4. Прозорість, відтворюваність та обґрунтованість. Весь процес дослідження фіксується в системі. Клієнт може отримати не лише фінальний звіт, а й:

– Трасування виконання: Відображення ланцюжка прийнятих рішень агента

(Scoping → Research → Synthesis);

- Список джерел: Детальна бібліографія з прямими посиланнями на використані статті, стандарти та звіти;
- Можливість верифікації: Це забезпечує академічну строгість та дозволяє технічним фахівцям перевірити першоджерела, що є критично важливим для R&D-відділів та прийняття архітектурних рішень.

Формати продукту пропонуються для різних потреб ринку:

- SaaS-платформа (підписка): Основний формат. Клієнти отримують доступ до веб-інтерфейсу або API, з можливістю формувати певну кількість запитів на місяць за різними тарифними планами (наприклад, Basic, Pro, Enterprise);
- коробкове рішення для внутрішнього розгортання (On-premise): Для клієнтів з підвищеними вимогами до конфіденційності даних (наприклад, військові заклади, великі вендори). Передбачає ліцензійне програмне забезпечення для встановлення на власній інфраструктурі клієнта;
- разові аналітичні замовлення (Custom Research): Послуга для виконання специфічних, нестандартних дослідницьких проектів з фіксованою вартістю за завдання.

Таким чином, продукт TRaaS є не просто інтерфейсом до LLM, а спеціалізованою інформаційно-аналітичною фабрикою, яка трансформує неструктурований інформаційний потік телекомунікаційної галузі в чіткий, дієвий аналітичний ресурс, економлячи найцінніші активи будь-якої організації — час експертів та фінансові ресурси.

4.3 Аналіз ринкових можливостей та цільових сегментів

Глобальний ринок телекомунікаційних послуг переживає період інтенсивної трансформації та зростання, що формує ідеальне середовище для запуску спеціалізованих сервісів на кшталт TRaaS. Очікується, що ринок зросте з приблизно 1.6 трлн дол. США у 2021 році до 2.55 трлн дол. США до 2031 року, демонструючи стійкий ріст. Це зростання живиться такими фундаментальними

факторами, як експоненційне збільшення мобільного трафіку даних, глобальне розгортання мереж 5G та масове впровадження технологій Інтернету речей (IoT). У таких умовах швидкість прийняття технологічних рішень стає критичним конкурентним перевагою, а обсяг інформації, яку необхідно аналізувати, значно перевищує можливості традиційних методів.

Сучасні тенденції галузі створюють конкретний, нагальний попит на автоматизовану аналітику:

1) технологічна складність та прискорення інновацій: перехід до хмарно-нативних архітектур (CNF), впровадження AI в радіодоступні мережі (AI-RAN), розвиток програмованих супутникових мереж (NTN) та стандартизація Open RAN роблять технологічний ландшафт надзвичайно складним. Компанії потребують постійного моніторингу цих змін для планування дорожніх карт та оцінки ризиків;
2) необхідність конкурентного та технологічного бенчмаркінгу: індустрія активно оцінює постачальників у різних сегментах — від антенних систем (DAS/DRS) до платформ телеком-API. Підготовка таких порівняльних аналізів вручну вимагає великих ресурсів часу;
3) спеціалізація штучного інтелекту: з'являються вертикальні LLM, спеціально навчені для телекомунікацій (наприклад, моделі Huawei, ZTE, SoftBank), що підтверджує тренд на глибоку галузеву адаптацію AI та відкриває нові можливості для автоматизації;
4) економічний тиск на R&D-відділи: прискорення циклів розробки (5G Advanced, 6G) поєднується з потребою оптимізації витрат. Інструмент, що дозволяє отримати технічний огляд за вартістю, що на порядок нижча за працю аналітика (порівняно з демонстрованими 0.4 USD), закриває критичну бізнес-потребу.

Продукт TRaaS розрахований на B2B-ринок, де прийняття рішень ґрунтується на технічній обґрунтованості та економічній доцільності. Сегменти розподілені за пріоритетністю та потенційним обсягом потреб.

Таблиця 4.1

Цільові сегменти клієнтів

Цільовий сегмент	Ключові представники	Болеві точки та потреби	Цінність TRaaS
Вендори телекомунікаційного обладнання	Ericsson, Nokia, Huawei, Cisco, ZTE, Juniper Networks.	Моніторинг конкурентів, аналіз ринкових ніш (напр., пасивні антени), технологічний бенчмаркінг, підготовка матеріалів для патентування, відстеження стандартів.	Швидке отримання структурованих звітів по конкурентах і трендах для стратегічного планування продуктів.
Оператори зв'язку	Verizon, Deutsche Telekom, China Mobile, Vodafone, AT&T.	Оцінка нових технологій (напр., 5G SA core, AI-RAN) перед інвестиціями, аналіз постачальників, дослідження нових моделей монетизації (напр., Telco GPUaaS).	Незалежний, об'єктивний аналітичний ресурс для прискорення прийняття архітектурних рішень.
Аналітичні та консалтингові агенції	ABI Research, Gartner (технічний сектор), спеціалізовані B2B-агентства.	Необхідність швидко готувати високоякісні, джерелами підкріплені ринкові звіти для клієнтів. Велика залежність від ручного аналізу.	Інструмент для значного прискорення дослідницького циклу та зниження собівартості підготовки звітів.
Академічні та дослідницькі інститути	Університетські лабораторії, державні наукові центри.	Проведення систематичних оглядів літератури (literature review) для наукових робіт, пошук актуальних джерел, генерація ідей для нових досліджень.	Демократизація доступу до комплексного аналізу, незамінний помічник для дослідників та аспірантів.
Корпоративні користувачі високих технологій	Великі підприємства з IoT-проектами, компанії з приватними мережами.	Оцінка телекомунікаційних рішень для власних потреб, розуміння ринку постачальників.	Спеціалізований експертний ресурс без необхідності наймати галузевого аналітика в штат.

Хоча ринок розвинений у Північній Америці та Європі, найдинамічніше зростання очікується в Азійсько-Тихоокеанському регіоні завдяки інтенсивному впровадженню 5G, IoT та смарт-технологій. Це робить регіон стратегічно важливим ринком для масштабування сервісу. Унікальна торгова пропозиція для всіх цих сегментів може бути сформульована так: «Автоматизований технічний аналітик з глибоким розумінням телекомунікацій: всебічний звіт за години, а не тижні, за вартістю, що на порядки нижча за експерта».

4.4 Конкурентний аналіз

Для сервісу Telecom Research Assistant as a Service (TRaaS) конкурентне середовище слід аналізувати через призму трьох основних категорій: прямі конкуренти, непрямі конкуренти (альтернативи) та продукти-замінники.

Категоризація конкурентів:

1) Прямі конкуренти: На сьогоднішній день на ринку відсутні готові комерційні рішення, що пропонують ідентичну послугу: повністю автономного, глибинного науково-технічного дослідження, спеціалізованого саме для телекомунікаційної галузі та оформленого у вигляді структурованого аналітичного звіту. Це свідчить про вільну нішу;
2) Непрямі конкуренти та альтернативи:
   - Загальні ШІ-асистенти (ChatGPT Plus, Claude, Gemini): Це основні альтернативи, які використовуються для пошуку інформації. Вони є потужними інструментами, але мають фундаментальні обмеження для професійного аналізу: відсутність галузевої спеціалізації, нездатність до автономного пошуку в реальному часі через авторитетні джерела (наприклад, IEEE Xplore, arXiv), обмежений контекстне вікно для обробки десятків документів та відсутність систематичного цитування джерел;
   - Професійні аналітики та консалтингові агенції (ABI Research, Gartner): Ці гравці пропонують високоякісні, але дуже дорогі послуги. Підготовка детального технічного звіту може коштувати десятки тисяч доларів і займати тижні. TRaaS позиціонується не як заміна експертному думці, а як інструмент для її значного прискорення та здешевлення на етапі первинного аналізу;
   - Спеціалізовані аналітичні платформи та бази даних: Рішення на кшталт ABI Research або YC.Market пропонують структуровані ринкові дані, звіти та бенчмарки. Однак це, як правило, пасивні джерела попередньо підготовленої інформації, тоді як TRaaS є активним інструментом, який генерує персоналізований аналіз на будь-який запит клієнта в реальному часі.

3) Конкуренти «на заміну»: До цієї категорії можна віднести внутрішніх співробітників компаній (інженери, аналітики), які виконують дослідницьку роботу вручну, а також традиційні методи роботи з літературою (самостійний пошук у наукових базах). Економічна перевага TRaaS тут є найбільш переконливою.

Наступна таблиця наочно демонструє позиціонування TRaaS серед основних альтернатив.

Таблиця 4.2

Порівняльний аналіз за ключовими критеріями

Критерій	**TRaaS**	**Загальні ШІ-асистенти (ChatGPT, тощо)**	**Професійні аналітики (ABI Research, консультанти)**	**Внутрішні фахівці (ручна робота)**
Галузева спеціалізація	Висока: адаптовані промпти, орієнтація на телеком-джерела.	Низька/відсутня: загальні знання, не розуміє галузевого контексту глибоко.	Дуже висока: експертність у конкретних сегментах ринку.	Залежить від кваліфікації
Глибина та автономність аналізу	Висока: автономне планування, пошук, синтез із десятків джерел.	Обмежена: відповідає на запит, але не проводить дослідження.	Дуже висока: глибинний аналіз, інтерпретація, прогнози.	Висока, але повільна
Швидкість отримання результату	Хвилини/години	Секунди/хвилини (на формулювання відповіді)	Тижні/місяці (на підготовку звіту)	Тижні
Вартість	Низька (центи за звіт)	Середня (підписки)	Дуже висока (тисячі/десятки тисяч доларів)	Висока (зарплата, навантаження)
Актуальність джерел	Висока: пошук у реальному часі через Tavily API, доступ до препринтів.	Обмежена датою відсічення знань моделі.	Висока (експертний моніторинг).	Залежить від зусиль співробітника
Цитування та відтворюваність	Систематичне (автоматичне формування списку джерел).	Відсутнє або неточне.	Високе (академічні стандарти).	Залежить від методики

Телекомунікаційна галузь знаходиться в стані трансформації, що формує як можливості, так і ризики для TRaaS:

– технологічні тренди: активний розвиток AI-RAN (AI у радіодоступних мережах), вертикальних LLM для телекомунікацій (наприклад, моделі Huawei, ZTE, SoftBank) та Open RAN свідчить про те, що інтеграція ШІ стає

основним напрямком. Це підтверджує актуальність продукту та може створити синергію (наприклад, використання спеціалізованих телеком-LLM як "мозку" в майбутньому);

– загроза входу нових гравців: існує ймовірність, що великі технологічні гіганти (Google, Microsoft, Amazon) або провідні вендори телекомобладнання (Ericsson, Nokia) можуть інтегрувати подібну функціональність у свої сервіси або платформи. Сильною стороною TRaaS у цьому випадку є фокус, швидкість ітерацій та глибока спеціалізація на одній задачі — генерації дослідницьких звітів;
– консолідація ринку: злиття та поглинання серед операторів зв'язку можуть скоротити кількість потенційних клієнтів, але одночасно збільшать потребу у ефективних інструментах для інтеграції та аналізу різних технологічних стеків.

Таблиця 4.3

SWOT-аналіз продукту TRaaS

Сильні сторони (Strengths)	**Слабкі сторони (Weaknesses)**
• Унікальна ніша: відсутність прямих конкурентів. • Екстремальна економічна ефективність: собівартість аналізу в десятки разів нижча за альтернативи. • Глибока галузева адаптація: система «розуміє» контекст телекомунікацій. • Швидкість: генерація комплексного звіту за години. • Прозорість: автоматичне цитування джерел та трасування логіки.	• Залежність від якості промптів та доступних моделей: результат залежить від точності налаштувань та конкретних LLM. • Відсутність критичної та прогностичної оцінки: система синтезує, але не критикує джерела глибоко. • Обмежена обробка конфліктуючих даних. • Низька впізнаваність бренду як нового гравця.

Аналіз конкурентного середовища підтверджує, що TRaaS займає унікальну та перспективну позицію на перетині двох трендів: автоматизації науково-технічного аналізу та галузевої спеціалізації штучного інтелекту. Продукт не має прямих аналогів, а в порівнянні з існуючими альтернативами демонструє переконливу перевагу за співвідношенням глибини, швидкості та вартості.

Основним викликом залишається не конкуренція з існуючими рішеннями, а потенційне з'явлення нових гравців та необхідність швидкої адаптації до стрімкої еволюції як телекомунікаційних технологій, так і самих моделей ШІ.

4.5 Бізнес-модель та стратегія виходу на ринок

Основою бізнес-моделі TRaaS є модель SaaS (Software as a Service), що забезпечує передбачуваний потік доходів, низький бар'єр для початку використання клієнтами та можливість швидкого масштабування. Унікальне співвідношення собівартості та цінності для клієнта дозволяє будувати гнучку багаторівневу структуру тарифів, орієнтовану на різні цільові сегменти. Модель має архітектуру "Freemium", що є ключовою для залучення перших користувачів та демонстрації цінності продукту.

Таблиця 4.4

Бізнес-моделі продукту TRaaS

Рівень / Модель	**Цільова аудиторія**	**Ключові можливості**	**Механізм монетизації**
Free Tier (Безкоштовний)	Окремі дослідники, студенти, академічні групи, для ознайомлення.	1-2 базові запити на місяць. Обмеження на обсяг звіту (напр., до 5 сторінок). Стандартна черга обробки. Базові теми.	Безкоштовно. Мета: формування спільноти, збір зворотного зв'язку, трансформація в платних користувачів.
Pro Tier (Професійний)	Малі та середні дослідницькі команди, стартапи в телеком-сфері, незалежні консультанти.	До 10-20 запитів на місяць. Розширений обсяг звітів (до 20-30 сторінок). Пріоритетна обробка. Доступ до розширених тем та шаблонів звітів. API доступ з лімітами.	Підписка від 99 USD/міс. Основний джерело доходу на ранній стадії.
Enterprise Tier (Корпоративний)	Великі вендори обладнання, оператори зв'язку, аналітичні агенції.	Необмежена кількість запитів або дуже високі ліміти. Пріоритетна підтримка 24/7. White-label звіти (брендування під клієнта). Повний API доступ для інтеграції в корпоративні системи. Персоналізація промптів під внутрішні стандарти.	Індивідуальна підписка (від 1500 USD/міс.) або корпоративна ліцензія (річна, від 15 000 USD/рік).

Економічним фундаментом моделі є глибока різниця між собівартістю та ціною. Якщо собівартість одного стандартного звіту складає ~0.4 USD (витрати на токени LLM через OpenRouter та пошукові API), то навіть при пакеті з 20 звітів за 99 USD (вартість ~4.95 USD за звіт для клієнта) рентабельність залишається дуже високою. Це дозволяє інвестувати в розвиток, маркетинг і підтримку, залишаючись конкурентноспроможним.

Стратегія виходу на ринок будується на принципі "від ранніх приймачів до масового ринку" (Crossing the Chasm) з акцентом на галузеву експертизу та сарафанне радіо в професійному середовищі.

Етап 1: Запуск та валідація (0-6 місяців)

Ціль: Запуск open beta-версії, валідація продукту в реальних умовах, формування ядра лояльних користувачів.

Дії:

- цільове залучення: безкоштовний доступ пропонується академічній спільноті (ВНЗ, наукові інститути), малим дослідницьким групам та технічним блогерам у сфері телекомунікацій;
- збір зворотного зв'язку: активна робота з ранніми користувачами для покращення промптів, інтерфейсу та логіки аналізу;
- формування кейсів: публікація разом з користувачами кейсів успішного використання (наприклад, "Як аспірант підготував огляд літератури по 6G за 1 день");
- канали: особисті зв'язки, професійні спільноти (LinkedIn, спеціалізовані форуми), партнерства з прогресивними науковими керівниками.

Етап 2: Активне просування та робота з ранніми B2B-клієнтами (6-18 місяців)

Ціль: Вихід на комерційні B2B-сегменти, формування стабленого потоку доходу, розширення команди.

Дії:

- публікація white-paper: офіційна публікація дослідження, що демонструє ефективність TRaaS у реальних умовах (порівняння

якості/швидкості/вартості з традиційними методами).

- цільова робота з ранніми B2B-адапторами: прямий Sales-підхід до технічних відділів середніх телеком-компаній, вендорів другого ешелону, аналітичних стартапів;
- розвиток партнерських програм: співпраця з консалтинговими агенціями, які можуть використовувати TRaaS як внутрішній інструмент для прискорення своїх проектів;
- контент-маркетинг: запуск професійного блогу, вебінарів з використання ШІ для досліджень у телекомунікаціях;
- канали: прямі продажі, партнерські мережі, B2B-конференції та відвідування телеком-заходів.

Етап 3: Масштабування та корпоративна інтеграція (18-36 місяців)

Ціль: Перетворення в стандартизований інструмент для галузі, вихід на великих корпоративних клієнтів.

Дії:

- прямі продажі великим вендорам та операторам: формування відділу продажів для роботи з лідерами ринку;
- глибока API-інтеграція: розробка рішень для безшовного впровадження TRaaS в корпоративні R&D та конкурентні розвідки;
- розширення функціоналу: на основі накопичених даних і запитів клієнтів — розвиток нових модулів (наприклад, автоматичний моніторинг патентів, аналіз тендерної документації);
- канали: прямі продажі через корпоративний Sales-відділ, участь у великих галузевих тендерах як постачальник технологій.

Успіх бізнес-моделі залежить від контролю над структурою витрат, яка на ранніх етапах буде складатися з (табл.4.5):

Таблиця 4.5

План структури витрат стартапу TRaaS

Категорія витрат	Основні статті	Коментар
Операційні витрати (OpEx)	Витрати на API (LLM через OpenRouter, Tavily Search, інші джерела).	Змінні витрати, що залежать від активності користувачів. Критично важливо оптимізувати через кешування, ефективні промпти.
Витрати на розробку та інфраструктуру	Хостинг, DevOps, зарплата AI-інженера/розробника Python.	Постійні витрати. Можна мінімізувати за рахунок використання хмарних сервісів з auto-scaling.
Витрати на продаж та маркетинг (S&M)	Контент-маркетинг, участь у заходах, зарплата sales-менеджера (на етапі 2).	Інвестиційні витрати для зростання. На етапі 1 зосередитись на низькобюджетних методах (SEO, соцмережі).
Адміністративні витрати (G&A)	Юридичні послуги, бухгалтерія, офісні витрати.	Мінімізуються за рахунок віддаленої роботи та аутсорсингу непрофільних функцій.

Прогноз доходів на перші три роки може бути консервативним: вихід на 50-100 платних підписок Pro на другому році та укладення 2-3 корпоративних контрактів Enterprise на третьому році вже забезпечить життєздатність та можливість подальшого масштабування.

Така бізнес-модель TRaaS, заснована на SaaS з підходом Freemium, поєднує в собі низький бар'єр для ознайомлення з високою рентабельністю за рахунок унікальної економіки продукту. Стратегія поступового, але цілеспрямованого виходу на ринок дозволяє мінімізувати ризики, сформувати лояльне ядро користувачів та послідовно завойовувати все більш платоспроможні сегменти ринку, починаючи з академічної спільноти і закінчуючи технологічними гігантами телекомунікаційної індустрії.

4.6 Практична реалізація

Перетворення дослідницького прототипу на комерційний продукт TRaaS вимагає системного погляду на технологічне вдосконалення, формування команди

та побудову правових та операційних підвалин.

Технологічна дорожня карта формується як послідовність кроків, кожен з яких вирішує конкретні обмеження прототипу та відкриває нові можливості. Початкова фаза (1-4 місяці) зосереджена на стабілізації ядра продукту: демонстраційний код перетворюється на індустріальну кодобазу з модульною архітектурою, суворим тестуванням та покращеною системою галузевих промптів. Ключовим завданням є підвищення надійності та передбачуваності роботи системи для перших бета-користувачів.

Наступна фаза (5-12 місяців) спрямована на створення повноцінної продуктивної платформи. Центральним технічним викликом тут є розробка механізму точного цитування, що дозволить прив'язувати твердження у звітах до конкретних розділів, сторінок та версій технічних стандартів, тим самим вирішуючи одне з головних виявлених обмежень. Паралельно відбувається перехід до хмарної, контейнеризованої інфраструктури з використанням мікросервісних підходів, що забезпечить можливість обробки сотень паралельних запитів. В цей же період розробляються публічне API та базовий веб-інтерфейс, які замінять для кінцевого клієнта інструменти розробника типу LangSmith Studio.

Довгострокова фаза розвитку (13-24 місяці) передбачає перехід від інструменту до платформи. Це включає можливість для корпоративних клієнтів інтегрувати власні, закриті корпуси документів для аналізу, що перетворить TRaaS на внутрішній дослідницький центр компанії. Архітектура системи розвиватиметься у бік гнучкої модульності, дозволяючи підключати різні LLM-провайдерів та аналітичні модулі залежно від конкретних потреб завдання, забезпечуючи таким чином майбутню конкурентоздатність та адаптивність.

Успіх технологічної дорожньої карти цілком залежить від мультидисциплінарної команди. На ранніх етапах ядро складатиметься з AI-розробника, який відповідає за архітектуру системи та логіку роботи з LLM, та телекомунікаційного експерта-предметника, чиї знання критично необхідні для валідації та налаштування галузевих промптів. У міру росту до команди приєднається backend- та DevOps-інженер для побудови масштабованої

інфраструктури, full-stack розробник для створення інтерфейсів клієнтів та product-менеджер, який забезпечить зв'язок між технологічними можливостями та ринковими потребами, формуючи roadmap продукту.

Створення комерційного сервісу також вимагає продуманого юридичного та етичного фундаменту. Першочерговим завданням є проведення аудиту ліцензій усіх використаних open-source компонентів (таких як ODR та LangChain) для гарантії їх сумісності з комерційним використанням. Умови надання послуг (Terms of Service) мають чітко визначати, що TRaaS є інструментом аналітичної підтримки, а не джерелом професійних консультацій, знімаючи таким чином юридичну відповідальність за рішення, прийняті на основі згенерованих звітів. Політика конфіденційності повинна бути прозорою щодо обробки метаданих та пропонувати корпоративним клієнтам опції повного контролю над даними, особливо в моделі on-premise розгортання. Система автоматичного цитування, окрім підвищення якості, слугуватиме і механізмом дотримання принципів добросовісного використання інтелектуальної власності.

Фінальним елементом пазлу є операційна модель, яка на ранніх етапах будується навколо гнучкої віддаленої роботи та agile-практик розробки. Критично важливими для зниження витрат та прискорення росту стануть стратегічні партнерства. Це включає переговори з постачальниками LLM-доступу про корпоративні тарифи, співпрацю з провайдерами спеціалізованих пошукових API для покращення якості джерел, а також альянси з технологічними консалтинговими агенціями, які можуть використовувати та рекомендувати TRaaS як частину своїх професійних послуг телеком-ринку.

Висновки

Аналіз та розробка стартап-проєкту на основі спеціалізованого ШІ-асистента для телекомунікацій демонструють життєздатну та інноваційну шлях трансформації академічної розробки в комерційно успішну бізнес-модель. Концептуальною основою проєкту є переосмислення ролі open-source рішення: від інструменту дослідження до ядра сервісної пропозиції, що посилює експертність клієнта за рахунок глибокої галузевої адаптації. Цей підхід, аналогічний моделі

таких компаній, як Red Hat, дозволяє сконцентруватися на унікальній цінності — наданні автоматизованого, всебічного технічного аналізу — без необхідності розробки базової інфракструктури з нуля. Експериментально підтверджена екстремальна економічна ефективність системи, де собівартість створення комплексного аналітичного звіту становить частки долара, формує фундаментальну конкурентну перевагу та робить послугу доступною для широкого спектра організацій.

Сформований продукт, «Telecom Research Assistant as a Service» (TRaaS), представляє собою не просто інтерфейс до великої мовної моделі, а спеціалізовану інформаційно-аналітичну фабрику. Його ядро цінності будується на чотирьох взаємопов’язаних стовпах: радикальній економії часу висококваліфікованих експертів, безпрецедентній економічній ефективності, «вбудованій» галузевій експертизі та повній прозорості та відтворюваності процесу дослідження. Така пропозиція закриває критичну потребу ринку в прискоренні та здешевленні процесів прийняття технологічних рішень у динамічній галузі телекомунікацій, де обсяг інформації та швидкість інновацій перевищують можливості традиційних методів.

Ринковий аналіз підтверджує наявність істотного попиту з боку ключових сегментів: вендорів телекомунікаційного обладнання, операторів зв’язку, аналітичних агенцій, академічних інститутів та корпоративних користувачів високих технологій. Кожен із цих сегментів має конкретні «болеві точки», які TRaaS здатний ефективно вирішити, пропонуючи швидкий, об’єктивний та технічно глибокий аналітичний ресурс. Конкурентне середовище характеризується відсутністю прямих аналогів, що відкриває унікальну нішу. У порівнянні з непрямими конкурентами — загальними ШІ-асистентами, професійними аналітиками та внутрішніми фахівцями — TRaaS демонструє переконливу перевагу за співвідношенням глибини аналізу, швидкості отримання результату, вартості та систематичності роботи з джерелами.

Запропонована бізнес-модель, заснована на принципах SaaS та Freemium, поєднує низький бар’єр для ознайомлення з високою рентабельністю завдяки

оптимальній структурі витрат. Трирівнева система тарифів (Free, Pro, Enterprise) дозволяє послідовно залучати різні категорії користувачів — від окремих дослідників до корпоративних гігантів. Стратегія виходу на ринок, розбита на три етапи — валідація з академічною спільнотою, активне просування серед B2B-клієнтів та масштабування через корпоративні інтеграції, — забезпечує поступове нарощування присутності та мінімізацію ризиків. Реалізація проєкту вимагає виконання чіткої технологічної дорожньої карти, формування мультидисциплінарної команди та побудови надійних юридичних та операційних підвалин, включаючи ліцензійну чистоту, прозорі умови надання послуг та механізми захисту конфіденційності даних.

Таким чином, стартап-проєкт TRaaS є не тільки логічним продовженням науково-дослідної роботи, але й життєздатною комерційною ініціативою, здатною трансформувати ландшафт прийняття рішень у телекомунікаційній індустрії. Він пропонує ринку спеціалізований інструмент, який посилює людську експертизу, роблячи комплексний технічний аналіз швидким, доступним і обґрунтованим. Подальший успіх проєкту залежатиме від здатності команди ефективно виконувати намічену стратегію, адаптуватися до стрімкої еволюції технологій ШІ та телекомунікацій, і, що найголовніше, неухильно забезпечувати високий рівень довіри та цінності для кожного клієнта.

ЗАГАЛЬНІ ВИСНОВКИ ПО РОБОТІ

Магістерська дисертація на тему «Використання технологій штучного інтелекту для проведення досліджень в сфері телекомунікацій» виконана відповідно до поставлених завдань. Робота присвячена розв'язанню актуальної проблеми автоматизації науково-аналітичної діяльності у динамічній та інформаційно насиченій галузі телекомунікацій.

В ході дослідження був проведений системний аналіз сучасних методів та інструментів автоматизації наукових досліджень. Встановлено, що традиційні підходи до пошуку та аналізу літератури не відповідають викликам, пов'язаним із експоненційним зростанням обсягів інформації, що актуалізує потребу у застосуванні ШІ-технологій. Проаналізовано архітектурні підходи та технології побудови спеціалізованих ШІ-агентів для наукового аналізу, що дозволило визначити ключові компоненти такий систем: концептуальну модель «Мислення-Дії-Спостереження», технології семантичного пошуку (RAG, векторні бази даних) та механізми динамічного оновлення знань. Порівняльний аналіз існуючих open-source платформ дозволив обґрунтовано обрати фреймворк Open Deep Research (ODR) як найбільш перспективну основу для побудови спеціалізованого агента в галузі телекомунікацій завдяки його модульності, орієнтації на глибинний аналіз та підтримці повного циклу дослідження.

Відповідно до обраного архітектурного рішення була розроблена та налаштована робоча система — спеціалізований ШІ-асистент для автоматизації наукового аналізу в телекомунікаціях. На базі ODR була реалізована адаптивна архітектура, що імітує ключові етапи наукового мислення (Scoping, Research, Synthesis) через станомісткий граф LangGraph. Критичною перевагою розробленої системи стала її орієнтація на пошук інформації в реальному часі через спеціалізовані API (Tavily Search), що забезпечило доступ до найновіших наукових публікацій, технічних стандартів і ринкових звітів. Практична валідація системи на прикладі комплексного запиту щодо архітектур 5G SA/NSA та вимог URLLC наочно підтвердила її працездатність, ефективність та економічну доцільність: собівартість генерації всебічного 20-сторінкового технічного звіту склала лише

частки долара.

Отриманий науково-технічний результат був трансформований у життєздатну бізнес-модель у рамках стартап-проєкту «Telecom Research Assistant as a Service» (TRaaS). Проєкт демонструє шлях комерціалізації академічної розробки через надання сервісу автоматизованого технічного аналізу. Запропонована бізнес-модель SaaS/Freemium, детальний аналіз ринкових сегментів, конкурентного середовища та стратегія поступового виходу на ринок обґрунтовують високу комерційну потенційність ініціативи. Система позиціонується не як заміна експерта, а як потужний синергетичний інструмент, що посилює людську експертизу, роблячи комплексний технічний аналіз швидким, доступним і обґрунтованим.

Практичним результатом роботи є функціонуючий спеціалізований ШІ-асистент, адаптований під предметну область, та комплексна бізнес-модель для його впровадження. Розробка має значний потенціал для масштабування та вдосконалення, зокрема шляхом покращення механізмів точного цитування, розвитку критичного мислення агентів та глибшої інтеграції в корпоративні R&D-процеси, що відкриває нові можливості для прискорення генерації знань і підвищення ефективності досліджень у телекомунікаційній галузі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Lewis, P. et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks”, Advances in Neural Information Processing Systems (NeurIPS), Vol. 33, 2020; [Електронний ресурс] – Режим доступу до ресурсу: https://arxiv.org/abs/2005.11401.
2. Open Deep Research Documentation [Електронний ресурс] // LangChain – Режим доступу до ресурсу: https://blog.langchain.com/open-deep-research.
3. Kairouz, P. et al., “Advances and Open Problems in Federated Learning”, Foundations and Trends in Machine Learning, Vol. 14, Видавництво Now Publishers, США, 2021; [Електронний ресурс] – Режим доступу до ресурсу: https://arxiv.org/abs/1912.04977.
4. McMahan, H.B. et al., “Communication-Efficient Learning of Deep Networks from Decentralized Data”, Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), Видавництво PMLR, США, 2017; [Електронний ресурс] – Режим доступу до ресурсу: https://arxiv.org/abs/1602.05629.
5. Russell, S.J., Norvig, P., “Artificial Intelligence: A Modern Approach. Classification of Intelligent Agents”, 4th Edition, Видавництво Pearson, США, 2020; [Друковане видання].
6. Algorithms and Architectures for Parallel Processing. ICA3PP 2022. Lecture Notes in Computer Science, Vol. 13727 / Eds. C.-Z. Xu et al., Видавництво Springer, Cham, 2023; [Електронний ресурс] – Режим доступу до ресурсу: https://link.springer.com/book/10.1007/978-3-031-22677-9.
7. What is Python? Executive Summary [Електронний ресурс] // Python Software Foundation – Режим доступу до ресурсу: https://www.python.org/doc/essays/blurb/.
8. LlamaIndex: The Data Framework for LLM Applications [Електронний ресурс] // LlamaIndex – Режим доступу до ресурсу: https://www.llamaindex.ai.
9. Wadhwani, J., “LlamaIndex vs LangChain: Which Framework is Best for Your LLM Application?”, Data Science Dojo Blog, 2024; [Електронний ресурс] –

Режим доступу до ресурсу: https://datasciencedojo.com/blog/llamaindex-vs-langchain/.

10. Haystack Documentation: Build Powerful NLP Search [Електронний ресурс] // deepset – Режим доступу до ресурсу: https://docs.haystack.deepset.ai/docs/intro.
11. ABI Research, “Telecommunications Market Research: Industry Analysis & Trends”, ABI Research, США, 2025; [Електронний ресурс] – Режим доступу до ресурсу: https://www.abiresearch.com/blog/telecommunications-market-research/.
12. Chen, J. et al., “ A Comprehensive Survey on Vector Database”, arXiv, 2023; [Електронний ресурс] – Режим доступу до ресурсу: https:// https://arxiv.org/abs/2310.11703.
13. Stodden, V. et al., “Enhancing reproducibility for computational methods”, Science, Vol. 354, Issue 6317, 2016; [Електронний ресурс] – Режим доступу до ресурсу: https://www.science.org/doi/10.1126/science.aah6168.
14. ChromaDB Documentation [Електронний ресурс] // Chroma – Режим доступу до ресурсу: https://docs.trychroma.com/.
15. IEEE P7009, “Standard for Fail-Safe Design of Autonomous and Semi-Autonomous Systems”, IEEE, США, 2021; [Електронний ресурс] – Режим доступу до ресурсу: https://standards.ieee.org/ieee/7009/7096/.
16. General Data Protection Regulation (GDPR) [Електронний ресурс] // European Union – Режим доступу до ресурсу: https://gdpr.eu/.
17. LangSmith Documentation [Електронний ресурс] // LangChain – Режим доступу до ресурсу: https://docs.langchain.com/langsmith/home.
18. GPT Researcher Official Website [Електронний ресурс] // gptr.dev – Режим доступу до ресурсу: https://gptr.dev.

ДОДАТКИ

Додаток А

Лістинг програмного модуля графу telecom_researcher.py

```
import asyncio
from typing import Literal
from langchain.chat_models import init_chat_model
from langchain_core.messages import AIMessage, HumanMessage,
SystemMessage, ToolMessage
from langgraph.graph import END, START, StateGraph
from langgraph.types import Command

# Імпорт ключових компонентів (скорочено)
from config import Configuration
from state import AgentState, SupervisorState, ResearcherState
from prompts import *  # Усі промпти імпортовані разом

class TelecomResearcher:
    """Головний клас для глибокого дослідження телекомунікацій."""

    def __init__(self):
        self.model = init_chat_model(configurable_fields=("model",
"api_key"))
        self.graph = self._build_graph()

    async def clarify_with_user(self, state: AgentState, config) ->
Command:
        """Уточнення завдання з користувачем."""
        cfg = Configuration.from_runnable_config(config)
        if not cfg.allow_clarification:
            return Command(goto="write_research_brief")

        # Аналіз потреби в уточненні
        response = await self._analyze_clarification(state, cfg)
        if response.need_clarification:
            return Command(goto=END, update={"messages":
[AIMessage(content=response.question)]})
        return Command(goto="write_research_brief")

    async def write_research_brief(self, state: AgentState, config)
-> Command:
        """Створення структурованого технічного завдання."""
        cfg = Configuration.from_runnable_config(config)
        response = await self._generate_research_brief(state, cfg)
        return Command(
            goto="research_supervisor",
            update={
                "research_brief": response.research_brief,
                "supervisor_messages":
[SystemMessage(content="Інструкції супервайзера"),
HumanMessage(content=response.research_brief)]
```

```
            }        )

    async def supervisor(self, state: SupervisorState, config) ->
Command:
        """Супервайзер дослідження - планування та делегування."""
        cfg = Configuration.from_runnable_config(config)
        response = await self._supervisor_decision(state, cfg)
        return Command(
            goto="supervisor_tools",
            update={"supervisor_messages": [response]})

    async def supervisor_tools(self, state: SupervisorState, config)
-> Command:
        """Виконання інструментів супервайзера."""
        cfg = Configuration.from_runnable_config(config)
        messages = state.get("supervisor_messages", [])

        # Умови завершення дослідження
        if self._should_end_research(state, cfg):
            return Command(goto=END,
update=self._prepare_end_state(state))

        # Обробка делегованих завдань
        tool_results = await self._process_research_tasks(state,
cfg)
        return Command(goto="supervisor", update=tool_results)

    async def researcher(self, state: ResearcherState, config) ->
Command:
        """Окремий дослідник - виконання конкретного завдання."""
        cfg = Configuration.from_runnable_config(config)
        tools = await self._get_research_tools(config)

        response = await self._researcher_work(state, cfg, tools)
        return Command(goto="researcher_tools",
update={"researcher_messages": [response]})

    async def researcher_tools(self, state: ResearcherState, config)
-> Command:
        """Інструменти дослідника (пошук, аналіз)."""
        cfg = Configuration.from_runnable_config(config)

        if self._should_compress_research(state, cfg):
            return Command(goto="compress_research")

        tool_results = await self._execute_researcher_tools(state,
config)
        return Command(goto="researcher", update=tool_results)

    async def compress_research(self, state: ResearcherState,
config):
        """Компресія результатів дослідження."""
        compressed = await self._synthesize_findings(state, config)
```

```
        return {
            "compressed_research": compressed,
            "raw_notes": [self._extract_notes(state)]
        }

    async def final_report_generation(self, state: AgentState,
config):
        """Генерація фінального звіту."""
        findings = "\n".join(state.get("notes", []))
        for attempt in range(3):
            try:
                report = await self._generate_report(state, config,
findings)
                return {
                    "final_report": report.content,
                    "messages": [report]
                }
            except Exception as e:
                if self._is_token_limit_error(e):
                    findings = findings[:int(len(findings) * 0.9)]
# Скорочення
                    continue
                raise
        return {"final_report": "Помилка генерації звіту"}

    # Допоміжні методи (скорочено)
    async def _analyze_clarification(self, state, cfg):
        """Аналіз потреби в уточненні."""
        pass

    async def _generate_research_brief(self, state, cfg):
        """Генерація техзавдання."""
        pass

    async def _supervisor_decision(self, state, cfg):
        """Прийняття рішень супервайзером."""
        pass

    def _should_end_research(self, state, cfg):
        """Перевірка умов завершення."""
        iterations = state.get("research_iterations", 0)
        return iterations > cfg.max_researcher_iterations

    async def _process_research_tasks(self, state, cfg):
        """Обробка паралельних досліджень."""
        tasks = []
        for task in state.get("research_tasks",
[])[:cfg.max_concurrent_research_units]:
            tasks.append(self._run_research_task(task, cfg))

        results = await asyncio.gather(*tasks)
        return {"supervisor_messages": results}
```

```
    async def _get_research_tools(self, config):
        """Отримання доступних інструментів."""
        return []

    async def _researcher_work(self, state, cfg, tools):
        """Робота дослідника."""
        pass

    def _should_compress_research(self, state, cfg):
        """Умови компресії результатів."""
        iterations = state.get("tool_call_iterations", 0)
        return iterations >= cfg.max_react_tool_calls

    async def _execute_researcher_tools(self, state, config):
        """Виконання інструментів дослідника."""
        pass

    async def _synthesize_findings(self, state, config):
        """Синтез результатів."""
        pass

    def _extract_notes(self, state):
        """Витяг нотаток."""
        return ""

    async def _generate_report(self, state, config, findings):
        """Генерація звіту."""
        pass

    def _is_token_limit_error(self, e):
        """Перевірка помилки ліміту токенів."""
        return "token" in str(e).lower()

    def _prepare_end_state(self, state):
        """Підготовка стану завершення."""
        return {
            "notes": state.get("notes", []),
            "research_brief": state.get("research_brief", "")}

    def _run_research_task(self, task, cfg):
        """Запуск окремого дослідження."""
        pass

    def _build_graph(self):
        """Побудова графа виконання."""
        builder = StateGraph(AgentState)

        # Додавання вузлів
        builder.add_node("clarify_with_user",
self.clarify_with_user)
        builder.add_node("write_research_brief",
self.write_research_brief)
        builder.add_node("research_supervisor",
```

```
self._build_supervisor_graph())
        builder.add_node("final_report_generation",
self.final_report_generation)

        # З'єднання вузлів
        builder.add_edge(START, "clarify_with_user")
        builder.add_edge("clarify_with_user",
"write_research_brief")
        builder.add_edge("write_research_brief",
"research_supervisor")
        builder.add_edge("research_supervisor",
"final_report_generation")
        builder.add_edge("final_report_generation", END)

        return builder.compile()

    def _build_supervisor_graph(self):
        """Побудова підграфа супервайзера."""
        builder = StateGraph(SupervisorState)
        builder.add_node("supervisor", self.supervisor)
        builder.add_node("supervisor_tools", self.supervisor_tools)
        builder.add_edge(START, "supervisor")
        builder.add_edge("supervisor", "supervisor_tools")
        builder.add_conditional_edges("supervisor_tools",
self._supervisor_routing)
        return builder.compile()

    def _build_researcher_graph(self):
        """Побудова підграфа дослідника."""
        builder = StateGraph(ResearcherState)
        builder.add_node("researcher", self.researcher)
        builder.add_node("researcher_tools", self.researcher_tools)
        builder.add_node("compress_research",
self.compress_research)
        builder.add_edge(START, "researcher")
        builder.add_edge("researcher", "researcher_tools")
        builder.add_conditional_edges("researcher_tools",
self._researcher_routing)
        builder.add_edge("compress_research", END)
        return builder.compile()

    def _supervisor_routing(self, state):
        """Маршрутизація супервайзера."""
        return "supervisor"  # Спрощено

    def _researcher_routing(self, state):
        """Маршрутизація дослідника."""
        if state.get("should_compress", False):
            return "compress_research"
        return "researcher"

# Глобальний екземпляр для використання
telecom_researcher = TelecomResearcher().graph
```