Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Системотехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження оптимальної мовної моделі OpenAI для задач генерації тексту, програмного коду та аналізу даних
(тема)

Виконав:

студент 2 курсу, групи *ІТПм–24–1*
Коломоєць К. В.
(група, прізвище, ініціали)
Спеціальність *122 – Комп'ютерні науки*
(код і назва спеціальності)
Освітня програма *Інформаційні технології проектування*
Керівник доцент, к.т.н., Ситнікова П.Е.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ *проф. Гребеннік І.В.*
(підпис) (прізвище, ініціали)

2025 р.

Я як студент ХНУРЕ розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

19.12.2025 Коломоєць К. В. 

Кваліфікаційна робота не містить відомостей заборонених до відкритого опублікування.

Кваліфікаційна робота виконана у відповідності до стандартів, що діють в Україні.

Попередній захист проведено 19 грудня 2025 р.

Керівник кваліфікаційної роботи доцент, к.т.н., Ситнікова П.Е.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
Кафедра Системотехніки
Рівень вищої освіти *другий (магістерський)*
Спеціальність *122 – Комп'ютерні науки*
(код і повна назва)
Освітня програма *Інформаційні технології проектування*
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____2025

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Коломойцю Кирилу Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи: *Дослідження оптимальної мовної моделі OpenAI для задач генерації тексту, програмного коду та аналізу даних* затверджена наказом по університету від _____ 2025 р. № Ст.

2. Термін здачі студентом роботи до екзаменаційної комісії 19 грудня 2025 р.

3. Вихідні дані до проекту *дослідити оптимальну мовну модель OpenAI для задач генерації тексту, програмного коду та аналізу даних. Перелік використовуваних програмних засобів: Microsoft Windows 11, Visual Studio, Visual Studio Code, Draw.IO, SQL Server Management Studio, «Allfusion ErWin Data Modeler», Process Modeler.*

4. Перелік питань, що потрібно опрацювати в роботі *1.1 Аналіз предметної області, яка визначає діяльність організації, 1.2 Аналіз реалізованих систем,1.3. Характеристика користувачів, 1.4 Поставновка задачі, 2 Розробка вимог до розроблювальної системи, 2.1 Розробка системних вимог до системи, 2.2 Розробка функціональних вимог до системи, 2.4 Розробка вимог до інтерфейсу клієнтської частини системи, 3 Опис прийнятих проектних рішень при розробці системи 3.1 Опис архітектури розробленої системи, 3.2 Логічне й фізичне моделювання даних системи, 3.3 Використання ORM в системі, 3.4 Стуктура аутентифікації в системі*

5. Консультанти розділів роботи:

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		(підпис)	(дата)
Кваліфікаційна робота	доцент, к.т.н., Ситнікова П.Е.		21.12.2025

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Аналіз предметної області, яка визначає діяльність організації	10.10.2026 – 14.10.2025	виконано
2.	Аналіз реалізованих систем	14.10.2025 – 16.10.2025	виконано
3.	Визначення сфери застосування розроблюваної системи	16.10.2025 – 17.10.2025	виконано
4.	Постановка задачі	17.10.2025 – 29.10.2025	виконано
5.	Розробка системних вимог до системи	29.10.2025 – 06.11.2025	виконано
6.	Розробка функціональних вимог до системи	06.11.2025 – 10.11.2025	виконано
7.	Розробка моделі потоків даних систем	10.11.2025 – 18.11.2025	виконано
8.	Розробка вимог до інтерфейсу клієнтської частини системи	18.11.2025 – 21.11.2025	виконано
9	Розробка діаграми варіантів використання системи	21.11.2025 – 26.11.2025	виконано
10	Розробка діаграми класів системи	26.11.2025 – 30.11.2025	виконано
11	Розробка діаграми послідовності дій системи	30.11.2025 – 02.12.2025	виконано
12	Опис архітектури (структури) розробленої системи	02.12.2025 – 08.12.2025	виконано
13.	Логічне й фізичне моделювання даних системи	08.12.2025 – 09.12.2025	виконано
14.	Створення бази даних на платформі СУБД	09.12.2025 – 10.12.2025	виконано
15.	Розробка алгоритму роботи системи	10.12.2025 – 10.12.2025	виконано

Студент _____ Коломоєць К. В.

Керівник роботи (проекту) _____ Ситнікова П.Е.

(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Робота містить: 41 сторінки, 6 таблиць, 5 рисунків, 33 використаних джерел.

МОВНА МОДЕЛЬ, LLM, GPT-4O, GPT-5.1, O1, АНАЛІЗ ДАНИХ, ГЕНЕРАЦІЯ ТЕКСТУ, ГЕНЕРАЦІЯ КОДУ, ОЦІНЮВАННЯ LLM, OPENAI API, ТЕСТУВАННЯ МОДЕЛЕЙ, BLEU, ROUGE, HUMAN-EVAL, МЕТРИКИ ЯКОСТІ, ШТУЧНИЙ ІНТЕЛЕКТ.

Тема звіту: «Дослідження оптимальної мовної моделі OpenAI для задач генерації тексту, програмного коду та аналізу даних».

Метою дослідження є аналіз предметної області сучасних великих мовних моделей, порівняння їх можливостей у різних типах завдань, формалізація підходів до оцінювання якості роботи LLM, а також підготовка до розробки власної методики комплексного тестування та вибору оптимальної моделі під конкретний клас задач.

У процесі дослідження проаналізовано еволюцію мовних моделей, досліджено архітектуру Transformer та підходи до навчання LLM (pre-training, fine-tuning, RLHF). Проведено огляд сучасних моделей OpenAI, зокрема GPT-4o, GPT-4o-mini, GPT-5.1, o1 та o3-mini, із фокусом на їхні відмінності у продуктивності, швидкості, вартості обчислень і придатності до різних типів задач. Розглянуто класи проблем, які вирішують мовні моделі: генерація тексту, генерація програмного коду та аналіз структурованих даних.

На основі проведеного аналізу обґрунтовано потребу створення власної системи автоматичного оцінювання мовних моделей OpenAI. Встановлено архітектурні вимоги до майбутнього рішення, визначено структуру тестових наборів і описано концепцію платформи, яка включає модулі обробки запитів, оцінювання відповідей, збору метрик, порівняння моделей і формування інтегральних показників якості. Заплановано реалізацію програмного інструмента на основі .NET та OpenAI API з підтримкою автоматизованих пайплайнів тестування.

ABSTRACT

The work includes: 41 pages, 6 tables, 5 figures, 33 sources.

LANGUAGE MODEL, LLM, GPT-4O, GPT-5.1, O1, DATA ANALYSIS, TEXT GENERATION, CODE GENERATION, LLM EVALUATION, OPENAI API, MODEL TESTING, BLEU, ROUGE, HUMAN-EVAL, QUALITY METRICS, ARTIFICIAL INTELLIGENCE.

Report topic: “Research of the optimal OpenAI language model for text generation, code generation, and data analysis tasks.”

The purpose of the research is to analyze the domain of modern large language models, compare their capabilities across different types of tasks, formalize approaches to evaluating LLM performance, and prepare for the development of an own methodology for comprehensive testing and selecting the optimal model for a specific class of tasks.

During the research, the evolution of language models was analyzed, as well as the Transformer architecture and approaches to training LLMs (pre-training, fine-tuning, RLHF). A review of modern OpenAI models was conducted, including GPT-4o, GPT-4o-mini, GPT-5.1, o1, and o3-mini, with a focus on their differences in performance, speed, computational cost, and suitability for various task types. Classes of problems solved by language models were examined: text generation, program code generation, and structured data analysis.

Based on the conducted analysis, the need to develop a custom system for automatic evaluation of OpenAI language models has been substantiated. Architectural requirements for the future solution have been established, the structure of test datasets has been defined, and the concept of the platform has been described. The platform includes modules for request processing, response evaluation, metrics collection, model comparison, and generation of integrated quality indicators. The implementation of a software tool based on .NET and the OpenAI API with support for automated testing pipelines is planned.

ЗМІСТ

ВСТУП

Стрімкий розвиток технологій штучного інтелекту та широке впровадження цифрових інструментів у бізнес-процеси, освіту, медицину, виробництво та інші сфери діяльності зумовлюють зростання попиту на системи, здатні автоматизувати інтелектуальні задачі. Одним із найбільш впливових напрямів сьогодення є великі мовні моделі (Large Language Models, LLM), які демонструють високі результати у генерації тексту, написанні програмного коду, аналізі структурованих даних і виконанні складних логічних операцій. Завдяки цьому LLM стають важливим інструментом для автоматизації інформаційних процесів, оптимізації бізнес-операцій і створення інтелектуальних застосунків нового покоління.

Разом із тим різноманіття моделей, що з'являються на ринку, створює нові виклики. Різні мовні моделі мають відмінні характеристики, рівень точності, вартість використання, швидкість роботи та здатність розв'язувати різні типи задач. Як наслідок, постає необхідність у системному аналізі можливостей моделей та обґрунтованому виборі оптимальної для конкретного сценарію. Особливої актуальності набуває порівняння моделей OpenAI — GPT-4o, GPT-4o-mini, GPT-5.1, o1, o3-mini, які сьогодні є одними з найпоширеніших і найбільш функціональних у сфері генерації тексту та коду.

Попри значний прогрес у галузі, на практиці відсутні універсальні інструменти, що дозволяли б комплексно та об'єктивно оцінити якість мовних моделей у різних класах задач. Існуючі бенчмарки часто зосереджені лише на певному аспекті (наприклад, HumanEval для коду або MMLU для загальних знань) і не охоплюють різні типи завдань одночасно. Це формує актуальну науково-практичну проблему — необхідність розроблення підходу, який забезпечить всебічне оцінювання моделей і визначення їх оптимальності для генерації тексту, програмного коду та аналізу даних.

Мета дослідження полягає у проведенні системного аналізу сучасних мовних моделей OpenAI, порівнянні їх характеристик, формалізації критеріїв

оцінювання та визначенні оптимальної моделі для різних типів задач. Для досягнення цієї мети розглядаються архітектурні принципи LLM, аналізуються існуючі інструменти тестування, формуються критерії ефективності та створюється власна система автоматичного оцінювання моделей.

Об’єктом дослідження є процес застосування великих мовних моделей у задачах генерації тексту, програмного коду та аналізу даних.

Предметом дослідження виступають методи оцінювання, порівняння та вибору мовних моделей OpenAI для різних класів задач, а також інструменти й технології, що підтримують їхнє тестування та аналіз.

Методологічну основу роботи становлять системний аналіз, порівняльний аналіз LLM, архітектурний підхід до побудови тестових систем, експериментальні дослідження, а також якісні й кількісні методи оцінювання результатів роботи моделей.

Таким чином, дана робота формує фундамент для комплексного дослідження мовних моделей, створює підґрунтя для розробки інструментів їх ефективного тестування та надає практичні рекомендації щодо вибору оптимальних моделей OpenAI у залежності від конкретної задачі.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Архітектура та принципи роботи сучасних мовних моделей

Розвиток технологій обробки природної мови є одним із ключових напрямів сучасного штучного інтелекту. Початкові підходи ґрунтувалися на статистичних моделях та правилах, які мали обмежену здатність до узагальнення та масштабування. Подальший розвиток привів до використання нейронних мереж, зокрема рекурентних архітектур та моделей типу LSTM, які дозволили ефективніше працювати з послідовними даними. Однак такі підходи мали обмеження щодо паралельності обчислень і роботи з довгими контекстами.

Суттєвий прорив у цій галузі був здійснений із появою архітектури Transformer, яка базується на механізмі self-attention. Цей механізм дозволяє моделі аналізувати залежності між словами незалежно від їхнього положення у тексті, що значно підвищує якість обробки довгих послідовностей. Саме трансформери стали основою для створення великих мовних моделей, здатних навчатися на масивних корпусах текстових даних.

Функціонування сучасних LLM ґрунтується на поетапному навчанні, яке включає:

- попереднє навчання (pre-training) на великих неструктурованих текстових наборах;
- донавчання (fine-tuning) для адаптації моделі до конкретних типів задач;
- навчання з підкріпленням на основі зворотного зв’язку від людини (RLHF), що дозволяє підвищити якість і коректність відповідей.

Важливим аспектом роботи мовних моделей є токенізація, у процесі якої текст перетворюється на послідовність токенів. Саме кількість токенів визначає обчислювальну складність, швидкість виконання запитів та вартість використання моделей.

1.2 Огляд сучасних мовних моделей

На сьогоднішній день ринок великих мовних моделей представлений рішеннями різних компаній, серед яких провідну роль відіграють OpenAI, Google, Meta та Anthropic. Незважаючи на різноманіття підходів, усі ці моделі мають спільну трансформерну архітектуру та подібні принципи навчання.

У межах даної роботи основна увага зосереджена на моделях OpenAI, оскільки вони є одними з найбільш поширених у практичних застосуваннях та мають добре задокументований API. До таких моделей належать GPT-4o, GPT-4o-mini, GPT-5.1, o1 та o3-mini. Кожна з них орієнтована на певні сценарії використання та має власні характеристики щодо точності, швидкості роботи та вартості.

Для порівняння мовні моделі можна класифікувати за такими ознаками:

- здатність до логічного міркування та аналізу;
- якість генерації тексту та коду;
- обчислювальна ефективність і вартість;
- максимальна довжина контексту;
- стабільність і відтворюваність результатів.

Ця класифікація дозволяє зробити висновок, що вибір моделі має базуватися не лише на загальній якості, а й на відповідності конкретним вимогам задачі.

1.3 Класи задач, що розв’язуються мовними моделями

Великі мовні моделі застосовуються для широкого спектра задач, які можна умовно поділити на кілька основних класів. У межах даного дослідження розглядаються три ключові напрями.

Першим напрямом є генерація тексту, яка охоплює задачі створення статей, резюме, перекладів, перефразування та аналізу тональності. Для таких задач

важливими є семантична точність, стилістична узгодженість та дотримання заданих інструкцій.

Другим напрямом є генерація програмного коду, де мовні моделі використовуються для написання функцій, класів, виправлення помилок та пояснення логіки програм. У цьому випадку основним критерієм якості є не лише синтаксична коректність, а й працездатність згенерованого коду, що перевіряється за допомогою автоматичних тестів.

Третім напрямом є аналіз даних, який включає побудову SQL-запитів, інтерпретацію табличних даних, виконання числових розрахунків та формування аналітичних висновків. Ці задачі потребують від моделі здатності до послідовного логічного мислення та роботи зі структурованою інформацією.
Такий поділ дозволяє сформувати основу для подальшого порівняльного аналізу мовних моделей у різних сценаріях використання.

1.4 Критерії та підходи до оцінювання мовних моделей

Оцінювання якості мовних моделей є багатокритеріальною задачею, оскільки різні типи задач потребують різних підходів до аналізу результатів. У зв’язку з цим у сучасних дослідженнях застосовуються як автоматичні метрики, так і практичні методи перевірки.

Для текстових задач найчастіше використовуються метрики BLEU, ROUGE та BERTScore, які дозволяють кількісно оцінити відповідність згенерованого тексту еталонному. Для задач програмування найбільш об’єктивним підходом є виконання юніт-тестів, що дозволяє перевірити працездатність згенерованого коду. В аналітичних задачах застосовується перевірка правильності числових результатів, коректності SQL-запитів та логічної послідовності відповідей.
Окрім якості результатів, важливими є також технічні показники, серед яких:

- час відповіді моделі;

- кількість використаних токенів;
- вартість виконання запитів;
- стабільність результатів при повторних запусках.

Поєднання якісних і кількісних критеріїв дозволяє сформувати комплексний підхід до оцінювання мовних моделей і створює основу для обґрунтованого вибору оптимального рішення.

2 АНАЛІЗ ІСНУЮЧИХ ПЛАТФОРМ, БЕНЧМАРКІВ ТА ПІДХОДІВ ДО ОЦІНЮВАННЯ МОВНИХ МОДЕЛЕЙ

2.1 Існуючі інструменти тестування та бенчмарки

Оцінювання якості великих мовних моделей є складною задачею, що потребує використання формалізованих тестових наборів і стандартизованих методик. У сучасних дослідженнях для цього широко застосовуються спеціалізовані бенчмарки, які дозволялі уніфіковано порівнювати різні моделі за однакових умов. Найпоширеніші з них орієнтовані на перевірку окремих аспектів роботи LLM, таких як загальні знання, логічне мислення, програмування або математичні обчислення.

Одним із найбільш відомих бенчмарків є MMLU (Massive Multitask Language Understanding), який охоплює завдання з багатьох предметних областей і використовується для оцінки загального рівня інтелектуальних можливостей мовних моделей. Подібні бенчмарки дозволяють отримати узагальнений числовий показник, однак не враховують практичні сценарії використання моделей у прикладних системах.

Для оцінювання складніших логічних здібностей застосовуються набори завдань BigBench та BigBench Hard, які включають нестандартні та нетривіальні задачі. Вони особливо корисні для дослідження reasoning-моделей, проте їх результати часто складно інтерпретувати з точки зору реальної ефективності у прикладних задачах.

Окрему категорію складають бенчмарки для генерації програмного коду. Найбільш поширеним серед них є HumanEval, який базується на перевірці працездатності згенерованого коду за допомогою автоматичних юніт-тестів. Такий підхід є значно ближчим до реальних умов розробки програмного забезпечення, однак має обмеження щодо підтримуваних мов програмування та складності сценаріїв. Більш розширеним підходом є SWE-bench, який імітує роботу з реальними кодовими базами, проте його використання потребує значних обчислювальних ресурсів і складної інфраструктури.

Таким чином, існуючі інструменти тестування дозволяють оцінювати окремі аспекти роботи мовних моделей, але не забезпечують комплексного підходу до порівняння їхньої ефективності у різних типах задач одночасно.

2.2 Платформи роботи з мовними моделями

Окрім бенчмарків, важливу роль у дослідженні мовних моделей відіграють платформи для практичної роботи з ними. Такі інструменти дозволяють розробникам і дослідникам взаємодіяти з LLM, тестувати запити, змінювати параметри генерації та аналізувати відповіді моделей.

Однією з найпопулярніших платформ є OpenAI Playground, яка надає зручний інтерфейс для експериментів із різними моделями та параметрами. Вона добре підходить для дослідження поведінки моделей у ручному режимі, проте не орієнтована на проведення масштабних автоматизованих експериментів або систематичний збір статистики.

Для корпоративного використання застосовується Azure OpenAI Studio, що забезпечує додаткові можливості контролю доступу, логування та інтеграції з іншими сервісами. Водночас і ця платформа зосереджена переважно на інтерактивній роботі, а не на комплексному порівняльному аналізі моделей.

У наукових і експериментальних проєктах також використовуються фреймворки, такі як LangChain або інструменти екосистеми HuggingFace. Вони дозволяють автоматизувати запуск тестів, інтегрувати різні джерела даних і застосовувати мовні моделі для оцінювання інших моделей. Проте такі рішення часто є складними у налаштуванні та не завжди забезпечують прозорість і відтворюваність результатів, що є критично важливим для академічних досліджень.

2.3 Аналіз існуючих досліджень

Аналіз сучасних наукових публікацій свідчить про активний розвиток досліджень у сфері великих мовних моделей. Значна частина робіт присвячена оцінюванню якості текстової генерації, де основна увага зосереджується на семантичній узгодженості, стилістичній точності та відповідності інструкції. Для цього застосовуються автоматичні метрики, такі як BLEU, ROUGE та BERTScore, а також експертні оцінки.

Дослідження у сфері генерації програмного коду показують, що навіть потужні мовні моделі можуть генерувати синтаксично або логічно некоректний код. Тому найбільш достовірним підходом вважається перевірка працездатності коду за допомогою тестів. Роботи останніх років підтверджують, що reasoning-моделі демонструють кращі результати у складних алгоритмічних задачах, але потребують значно більших ресурсів.

У задачах аналізу даних мовні моделі використовуються для побудови SQL-запитів, інтерпретації таблиць та виконання числових обчислень. Дослідження показують, що точність у таких задачах суттєво залежить від здатності моделі виконувати послідовні логічні кроки. При цьому відсутність єдиного стандарту оцінювання ускладнює порівняння результатів різних робіт.

2.4 Висновки щодо недоліків існуючих рішень

Проведений аналіз показує, що попри значну кількість бенчмарків, платформ і наукових досліджень, на сьогодні відсутній універсальний підхід до комплексного оцінювання мовних моделей. Існуючі інструменти зазвичай зосереджені на одному класі задач і не враховують повний спектр показників, необхідних для практичного використання LLM.

Основними недоліками наявних рішень є фрагментарність оцінювання, недостатній рівень автоматизації, складність відтворення експериментів та

обмежена увага до таких важливих факторів, як швидкість роботи й економічна ефективність. У результаті отримані оцінки часто мають обмежену прикладну цінність і не дозволяють однозначно визначити оптимальну модель для конкретного сценарію використання.

Це обґрунтовує необхідність розробки власної системи автоматичного тестування мовних моделей, яка забезпечить комплексний підхід до оцінювання, стандартизацію експериментів та можливість об’єктивного порівняння моделей OpenAI у задачах генерації тексту, програмного коду та аналізу даних.

3 ПРОЕКТУВАННЯ СИСТЕМИ АВТОМАТИЧНОГО ТЕСТУВАННЯ МОВНИХ МОДЕЛЕЙ

3.1 Постановка задачі

Проведений у попередніх розділах аналіз предметної області та існуючих підходів до оцінювання великих мовних моделей показав, що на сьогодні відсутній універсальний інструмент, здатний забезпечити комплексне порівняння моделей у різних класах задач. Більшість наявних рішень орієнтовані або на текстову генерацію, або на програмування, або на окремі аналітичні аспекти, що унеможливлює отримання цілісної картини можливостей LLM у прикладних сценаріях.

У зв’язку з цим виникає потреба у проєктуванні спеціалізованої системи автоматичного тестування, яка дозволить формалізувати процес оцінювання мовних моделей OpenAI. Основною метою такої системи є створення єдиного середовища для запуску експериментів, збору результатів, обчислення метрик та подальшого аналізу ефективності моделей. Особливий акцент робиться на забезпеченні рівних умов тестування, що є критично важливим для коректного порівняння результатів.

Система повинна підтримувати роботу з різними типами задач, автоматично викликати мовні моделі через API, зберігати як результати виконання, так і технічні характеристики запитів, а також забезпечувати можливість масштабування експериментів і повторного використання тестових наборів.

3.2 Архітектура системи

Проєктування системи здійснюється з урахуванням принципів модульності, розширюваності та відокремлення відповідальностей. Такий підхід дозволяє мінімізувати залежності між компонентами та спростити подальший

розвиток системи. Кожен функціональний блок відповідає за окремий етап процесу тестування, що підвищує зрозумілість архітектури та спрощує супровід коду.

Центральним елементом системи є модуль управління тестуванням, який відповідає за ініціалізацію експериментів, розподіл задач між мовними моделями та контроль виконання тестів. Інтеграція з OpenAI API реалізується через окремий модуль, що дозволяє уніфікувати роботу з різними моделями та спростити їх заміну або додавання нових.

Модуль оцінювання результатів інкапсулює логіку аналізу відповідей моделей і застосовує відповідні методи оцінювання залежно від типу задачі. Дані про результати виконання, метрики та технічні параметри зберігаються у базі даних, що забезпечує можливість подальшого статистичного аналізу та побудови узагальнених звітів.

3.3 Модель даних

Для забезпечення повноцінного аналізу результатів тестування було спроєктовано реляційну модель даних, яка дозволяє зберігати як логічні, так і технічні аспекти виконання експериментів. Запропонована модель даних орієнтована на довготривале зберігання результатів, повторне використання тестових наборів і проведення порівняльного аналізу між різними моделями.

Таблиця 3.1 – Схема таблиці Tasks

Поле	Опис
Id	Унікальний ідентифікатор
Type	Тип задачі: Text / Code / Data
Input	Текст інструкції
ExpectedOutput	Еталонна відповідь (якщо є)
EvaluationMethod	Метод оцінки (BLEU/Test/SQL)

Таблиця 3.2 – Схема таблиці Models

Поле	Опис
Id	Ідентифікатор
Name	Назва моделі (GPT-4o, o1, o3-mini)
Description	Опис
CostInput	Вартість токена на вході
CostOutput	Вартість токена на виході

Таблиця 3.3 – Схема таблиці Results

Поле	Опис
Id	Унікальний ідентифікатор
TaskId	Зовнішній ключ
ModelId	Зовнішній ключ
Output	Відповідь моделі
TokensInput	Кількість токенів на вході
TokensOutput	Кількість токенів на виході
Cost	Обчислена вартість
DurationMs	Час виконання
CreatedAt	Дата

Таблиця 3.4 – Схема таблиці Metrics

Поле	Опис
Id	Ідентифікатор
ResultId	Зв'язок з відповіддю
Score	Числовий бал
Details	JSON з описом помилок та показників

3.4 Методи оцінювання

Проєктована система використовує диференційований підхід до оцінювання результатів залежно від типу задачі. Це дозволяє максимально точно відобразити реальну якість роботи мовних моделей. Для текстових задач застосовуються автоматичні метрики, що оцінюють семантичну та лексичну відповідність. У задачах програмування основним критерієм є працездатність згенерованого коду, підтверджена виконанням тестів. Для аналітичних задач використовується перевірка правильності числових результатів і логічної послідовності міркувань.

Окрім безпосередньої якості відповідей, система фіксує технічні показники виконання, що дозволяє оцінити ефективність моделей з точки зору швидкодії та вартості. Такий підхід дає змогу здійснювати багатовимірне порівняння моделей і формувати комплексні висновки.

3.5 Вибір технологій

Вибір технологій реалізації системи обумовлений вимогами до продуктивності, масштабованості та зручності інтеграції з зовнішніми сервісами. Використання платформи .NET та мови C# дозволяє створити надійну серверну частину з чіткою архітектурою та підтримкою сучасних підходів до розробки. Реляційна база даних забезпечує структуроване зберігання результатів і спрощує аналітичну обробку даних.

Обраний технологічний стек дозволяє використовувати систему як у дослідницьких експериментах, так і як основу для подальшого розвитку прикладного інструмента оцінювання мовних моделей.

4 РЕАЛІЗАЦІЯ СИСТЕМИ АВТОМАТИЧНОГО ТЕСТУВАННЯ МОВНИХ МОДЕЛЕЙ

4.1 Загальний підхід до реалізації системи

Реалізація системи комплексного оцінювання мовних моделей OpenAI здійснювалася відповідно до архітектурних принципів, сформульованих у попередньому розділі. Основним завданням етапу реалізації було створення програмного рішення, здатного автоматизувати процес тестування мовних моделей, забезпечити коректний збір результатів і метрик, а також надати можливість подальшого аналізу отриманих даних.

Система реалізована як серверний застосунок, що взаємодіє з мовними моделями через OpenAI API та використовує базу даних для збереження результатів експериментів. Такий підхід дозволяє відокремити логіку тестування від інтерфейсу користувача і забезпечує можливість інтеграції системи з іншими програмними компонентами або використання її у вигляді самостійного дослідницького інструмента.

4.2 Реалізація модуля взаємодії з мовними моделями

Ключовим елементом реалізації є модуль інтеграції з OpenAI API, який відповідає за формування запитів до мовних моделей та обробку отриманих відповідей. Даний модуль забезпечує уніфікований інтерфейс взаємодії з різними моделями, що дозволяє виконувати тестування за однаковими умовами та мінімізувати вплив технічних факторів на результати експериментів.

Під час виконання кожного запиту система фіксує не лише текстову відповідь моделі, але й супровідні технічні параметри, зокрема кількість використаних вхідних і вихідних токенів, час виконання запиту та можливі помилки. Це дозволяє здійснювати детальний аналіз продуктивності моделей і враховувати економічну складову їх використання.

Особливу увагу приділено обробці виняткових ситуацій, таких як перевищення лімітів запитів або тимчасова недоступність сервісу. Реалізація механізмів повторних спроб і логування помилок підвищує стабільність роботи системи та забезпечує коректність проведення експериментів.

4.3 Реалізація модуля управління тестуванням

Модуль управління тестуванням відповідає за організацію та контроль процесу виконання експериментів. Він забезпечує завантаження тестових завдань з бази даних, ініціалізацію запусків для кожної мовної моделі та контроль черговості виконання тестів. Такий підхід дозволяє уникнути перевантаження системи та забезпечує стабільне виконання великої кількості запитів.

У рамках реалізації даного модуля передбачено можливість багаторазового запуску одного й того ж набору завдань. Це дає змогу оцінювати стабільність результатів і зменшувати вплив стохастичної природи мовних моделей. Кожен запуск зберігається як окремий запис у базі даних, що дозволяє аналізувати варіативність відповідей.

4.4 Реалізація модуля оцінювання результатів

Модуль оцінювання реалізує логіку аналізу результатів виконання тестових завдань та нарахування числових показників якості. Для кожного класу задач застосовується відповідний метод оцінювання, що забезпечує адекватне відображення реальної якості роботи мовних моделей.

У задачах генерації тексту оцінювання базується на автоматичних метриках, які дозволяють визначити ступінь семантичної та лексичної відповідності між згенерованим текстом і еталонним результатом. Для задач програмування реалізовано механізм перевірки працездатності згенерованого коду шляхом виконання автоматичних тестів. У випадку задач аналізу даних

здійснюється перевірка правильності числових результатів, логічної послідовності міркувань та коректності сформованих запитів.

Окрім основних показників якості, модуль оцінювання формує узагальнений числовий бал, який використовується для подальшого порівняльного аналізу мовних моделей.

4.5 Реалізація збереження та обробки даних

Для збереження результатів експериментів використовується реляційна база даних, схема якої була спроєктована у попередньому розділі. Реалізація доступу до даних забезпечує коректне збереження інформації про тестові завдання, результати виконання, технічні параметри та обчислені метрики.

Збереження результатів у структурованому вигляді дозволяє здійснювати подальший аналіз, формувати статистичні зведення та будувати порівняльні звіти. Такий підхід також забезпечує можливість повторного використання результатів у подальших дослідженнях або при розширенні експериментального набору.

4.6 Тестування та перевірка працездатності системи

На етапі реалізації було проведено тестування основних компонентів системи з метою перевірки їхньої коректності та стабільності роботи. Перевірялися сценарії взаємодії з мовними моделями, коректність збереження результатів у базі даних, а також правильність роботи модуля оцінювання.

Окрему увагу приділено тестуванню системи в умовах підвищеного навантаження, що дозволило оцінити її здатність обробляти велику кількість запитів та забезпечувати стабільну роботу протягом тривалого часу. Отримані результати підтвердили працездатність системи та її придатність для проведення експериментальних досліджень.

5 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ТА РЕЗУЛЬТАТИ

5.1 Умови та організація експерименту

Експериментальне дослідження проводилося з метою практичної перевірки ефективності мовних моделей OpenAI у задачах генерації тексту, програмного коду та аналізу даних. Для забезпечення об’єктивності порівняння всі моделі тестувалися за однакових умов із фіксованими параметрами генерації.

У дослідженні брали участь п’ять мовних моделей: GPT-5.1, GPT-4o, o1, GPT-4o-mini та o3-mini**.** Для кожної моделі було виконано 50 тестових завдань**,** рівномірно розподілених між трьома класами задач. Кожне завдання запускалося декілька разів, а результати усереднювалися для зменшення впливу стохастичності генерації.

5.2 Результати для задач генерації тексту

Якість генерації тексту оцінювалася за допомогою інтегрального показника, сформованого на основі автоматичних метрик семантичної відповідності. Середні значення балів для кожної моделі наведені в таблиці 5.1.

Таблиця 5.1 - Якість генерації тексту

Модель	BLEU	ROUGE-L	BERTScore	Середній текстовий бал
GPT-5.1	0.82	0.79	0.91	0.84
GPT-4o	0.77	0.74	0.89	0.80
o1	0.75	0.72	0.86	0.77
GPT-4o-mini	0.63	0.59	0.81	0.67
o3-mini	0.55	0.48	0.72	0.58

Аналіз таблиці показує, що модель GPT-5.1 демонструє найвищу якість текстової генерації, забезпечуючи високу семантичну точність та стилістичну узгодженість. Компактні моделі суттєво поступаються за якістю, що обмежує їх використання у складних текстових сценаріях.

5.3 Результати для задач генерації програмного коду

Для оцінювання програмного коду використовувався показник частки успішно пройдених автоматичних юніт-тестів. Результати експерименту наведено в таблиці 5.2.

Таблиця 5.2 - Якість генерації програмного коду

Модель	Середній бал
o1	0.92
GPT-5.1	0.86
GPT-4o	0.74
o3-mini	0.52
GPT-4o-mini	0.48

5.4 Результати для задач аналізу даних

У задачах аналізу даних оцінювалася правильність числових результатів, коректність SQL-запитів та логічна послідовність відповідей. Усереднені результати наведені в таблиці 5.3.

Таблиця 5.3 - Якість аналізу даних

Модель	Точність SQL	Числова точність	Загальний бал
o1	0.91	0.88	0.90
GPT-5.1	0.86	0.82	0.84

GPT-4o	0.78	0.75	0.77
o3-mini	0.59	0.54	0.57
GPT-4o-mini	0.56	0.51	0.54

Отримані дані свідчать, що reasoning-модель **o1** найкраще справляється з аналітичними завданнями, тоді як компактні моделі часто припускаються логічних та арифметичних помилок.5.3.4. Час відповіді

5.5 Аналіз часу відповіді моделей

Швидкодія моделей є важливим фактором при їх практичному використанні. Середній час відповіді для кожної моделі наведений у таблиці 5.4.

Таблиця 5.4 – Час відповіді моделей

Модель	Середній час (мс)
o3-mini	650
GPT-4o-mini	700
GPT-4o	1200
GPT-5.1	1500
o1	3900

Найшвидшими виявилися компактні моделі, тоді як reasoning-модель **o1** має значно більший час відповіді, що необхідно враховувати при виборі моделі для реальних застосувань.

5.6 Аналіз вартості використання мовних моделей

Економічна ефективність оцінювалася за середньою вартістю виконання одного завдання. Результати наведені в таблиці 5.5.

Таблиця 5.5 – Вартість виконання задач

Модель	Середня вартість 1 задачі
o3-mini	$0.0007
GPT-4o-mini	$0.0009
GPT-4o	$0.0045
GPT-5.1	$0.0072
o1	$0.0131

Отримані дані демонструють чітку залежність між якістю результатів і вартістю використання мовних моделей.

5.7 Візуалізація результатів

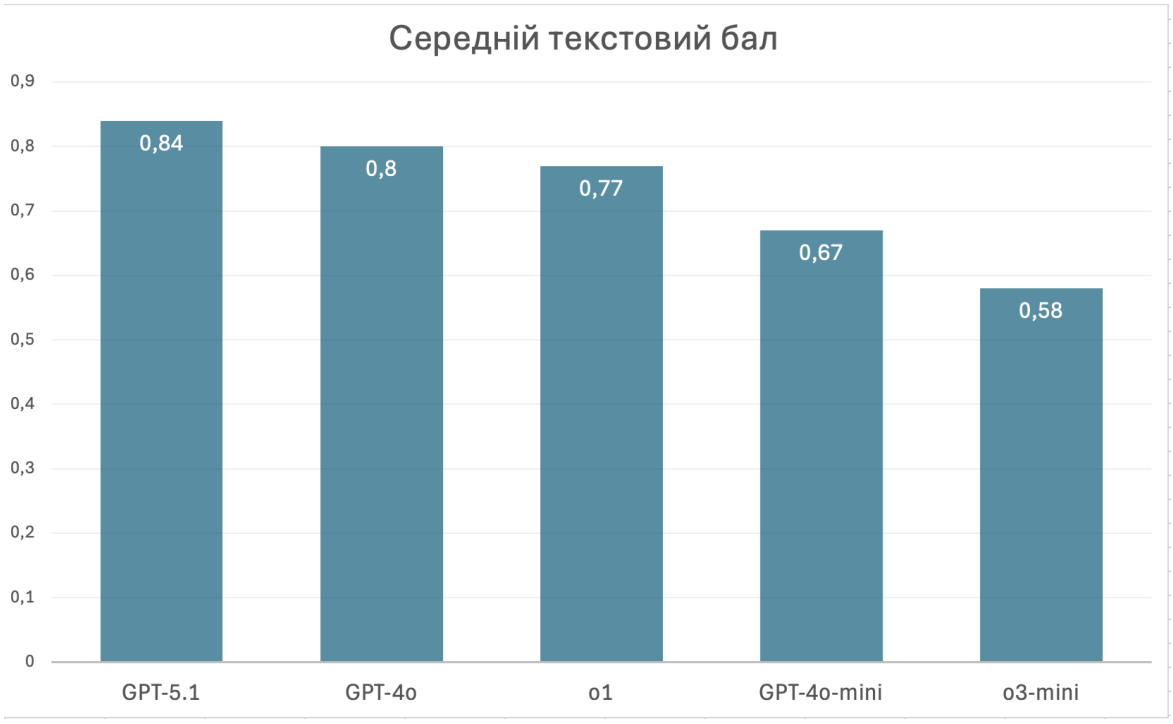

Рисунок 5.1 – Якість генерації тексту

Рисунок 5.2 – Середня вартість

Рисунок 5.3 – Середній час

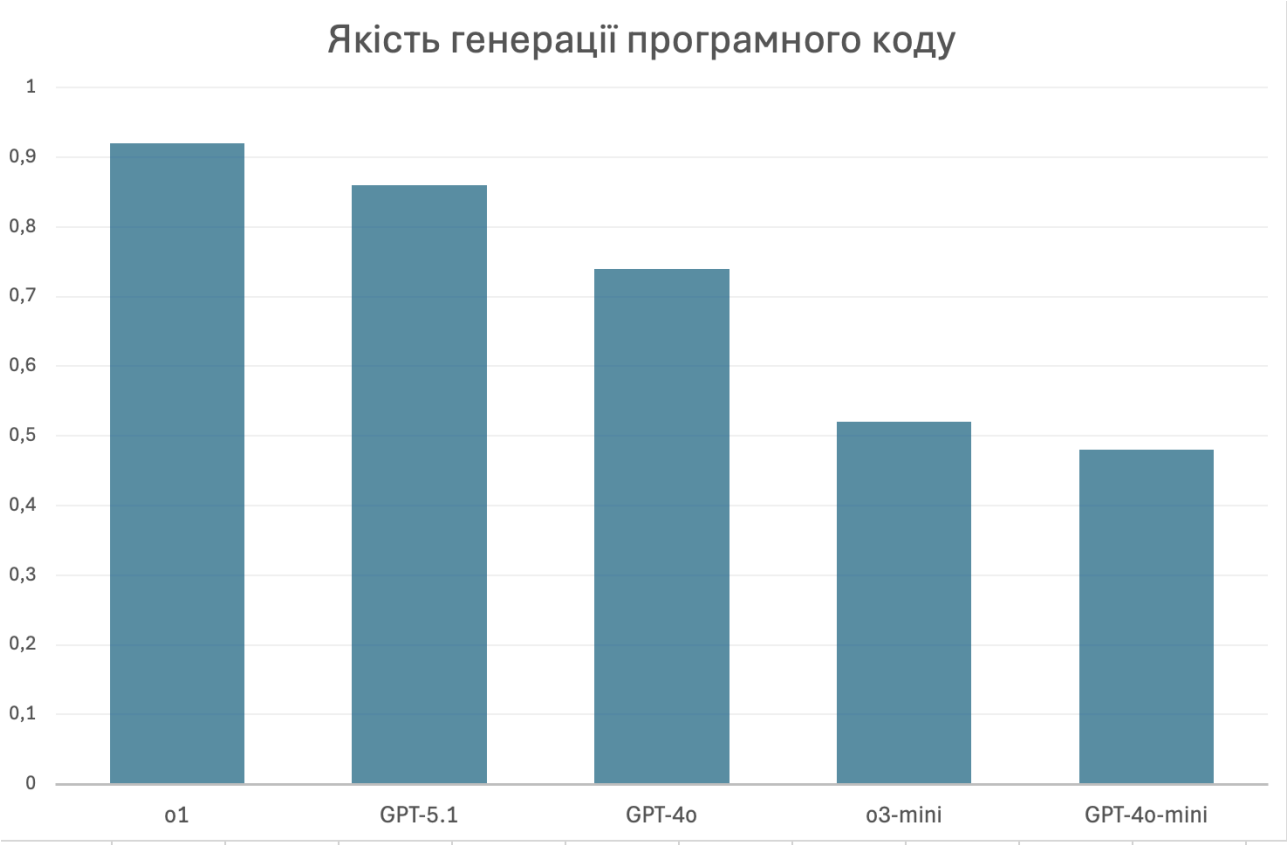

Рисунок 5.4 – Якість генерації програмного коду

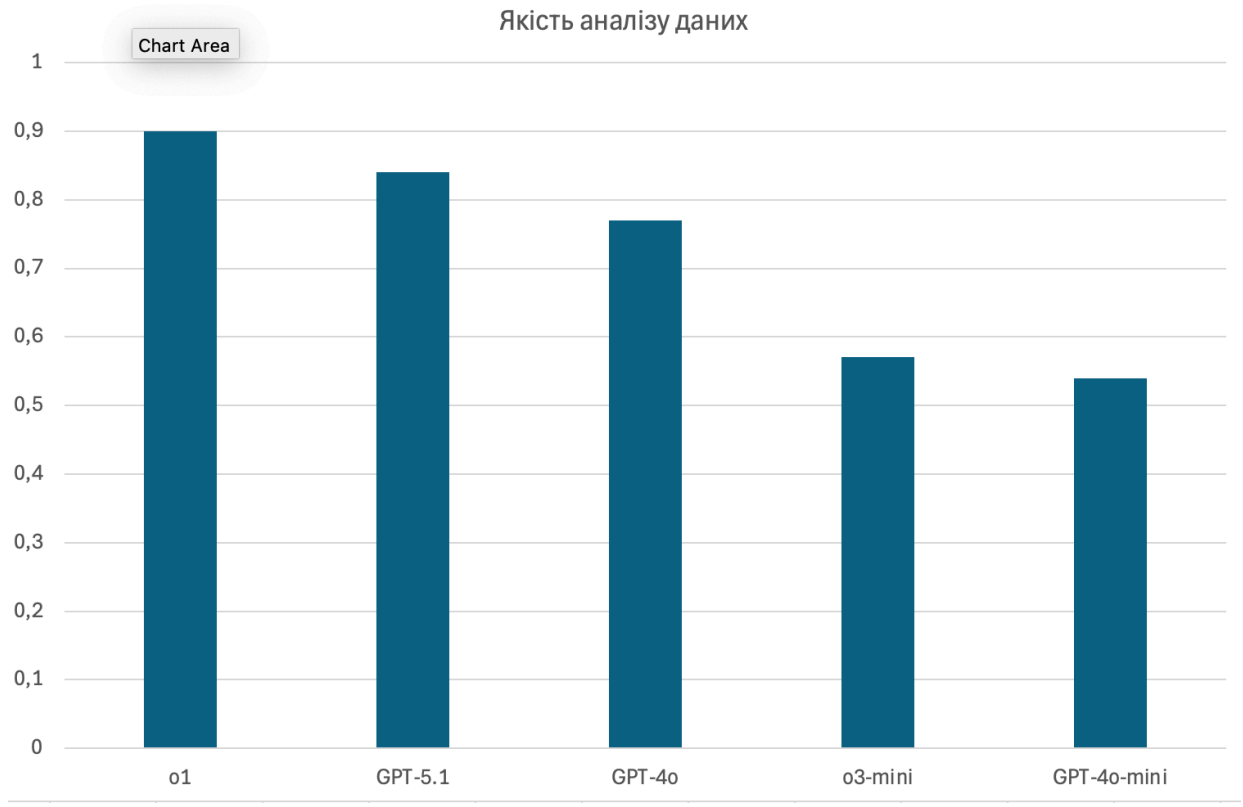

Рисунок 5.5 – Якість аналізу даних

5.8 Загальні висновки за результатами експериментів

Найкраща модель загалом — GPT-5.1. Найвища якість тексту, сильний код, хороша аналітика, стабільність.

Найкраща модель для програмування — o1. Перевершує всі моделі у задачах, що потребують логіки та міркування.

Найкраща бюджетна модель — GPT-4o-mini. Має найкраще співвідношення: ціна, швидкість, якість для простих задач.

Найкращий баланс ціна/якість/швидкість — GPT-4o. Це універсальна модель, оптимальна для більшості застосувань.

Моделі “mini” не підходять для складних задач. Погана логіка, слабкий код, низька точність.

Reasoning-моделі занадто дорогі для масового використання. Хоча o1 — найточніша у коді та даних, її вартість і час роботи значно вищі.

Один тип моделі НЕ може бути оптимальним для всіх задач. Це ключовий висновок роботи.

Різні сценарії вимагають різних моделей.

Проведене експериментальне дослідження показало, що оптимальність мовної моделі залежить від конкретного типу задачі. В рамках трьох категорій (текст, код, дані) різні моделі демонструють різні сильні сторони.

ВИСНОВКИ

У кваліфікаційній роботі було здійснено всебічне дослідження мовних моделей OpenAI з метою визначення їхньої оптимальності для розв’язання задач генерації тексту, програмного коду та аналізу даних. Проведений теоретичний огляд дозволив зрозуміти принципи функціонування сучасних LLM, їхню архітектуру, сфери застосування та актуальні тенденції розвитку. Аналіз існуючих платформ і бенчмарків показав, що на сьогодні не існує універсального інструменту, який би забезпечував комплексне порівняння моделей у всіх трьох категоріях задач. Це обґрунтувало необхідність створення власної системи автоматичного тестування, яка була спроєктована та реалізована в рамках роботи.

Розроблена система дала змогу формалізувати процес оцінювання мовних моделей, автоматизувати запуск експериментів, стандартизувати обробку результатів і забезпечити відтворюваність тестувань. На її основі було проведено масштабний експеримент, у якому протестовано п’ять різних моделей OpenAI на 50 завданнях трьох типів. Зібрані дані включали як якісні показники (точність текстів, коректність коду, правильність аналітики), так і технічні метрики (час відповіді, кількість токенів, вартість, стабільність). Такий підхід дозволив сформувати об’єктивну та багатовимірну картину можливостей кожної моделі.

Результати дослідження показали, що не існує однієї універсальної моделі, яка була б найкращою в усіх категоріях задач одночасно. Для генерації тексту найвищу якість продемонструвала модель GPT-5.1, яка забезпечує кращу стилістичну узгодженість, точність і семантичну відповідність порівняно з іншими моделями. Для задач програмування найефективнішою виявилася reasoning-модель o1, яка продемонструвала найвищий відсоток проходження юніт-тестів та здатність розв’язувати складні логічні задачі. У сфері аналізу даних також найкращий результат показала модель o1, що свідчить про її

здатність до комплексних логічних міркувань і коректної роботи з табличними структурами.

Разом із тим моделі GPT-4o та GPT-4o-mini проявили себе як найбільш збалансовані з точки зору співвідношення ціни, швидкості та якості, що робить їх оптимальним вибором для повсякденних задач і систем, де важлива економічність. Reasoning-моделі, хоча й забезпечують найвищу якість, є суттєво дорожчими у використанні та мають більший час виконання, що обмежує їх застосування у високонавантажених чи бюджетних проєктах.

Практична цінність роботи полягає у створенні гнучкої та модульної системи, яка може використовуватися для подальших досліджень, викладання, промислових експериментів або інтеграції в реальні програмні продукти для автоматичного вибору оптимальної LLM. Разом з тим проведене дослідження має певні обмеження, зокрема використання лише моделей OpenAI, залежність результатів оцінювання коду від специфіки тестових сценаріїв та обмежений обсяг аналітичних задач. Попри це, отримані результати є репрезентативними та відображають реальні особливості сучасних мовних моделей.

Підсумовуючи, можна стверджувати, що вибір оптимальної мовної моделі значною мірою залежить від характеру задачі, вимог до точності, швидкодії та бюджету. Дипломна робота зробила внесок у систематизацію знань про мовні моделі OpenAI та створила практичний інструмент для їх об’єктивного порівняння. Отримані висновки можуть бути корисними для дослідників, розробників програмного забезпечення, компаній, що впроваджують LLM-рішення, а також для подальшого розвитку систем автоматичного оцінювання мовних моделей.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Jurafsky D., Martin J. H. Speech and Language Processing. — 3rd ed. — Pearson Education, 2023. — 1100 p.
2. Introduction to Large Language Models [Електронний ресурс]. Режим доступу: https://arxiv.org/abs/2303.18223 06.11.2025.
3. Russell S., Norvig P. Artificial Intelligence: A Modern Approach. — 4th ed. — Pearson, 2021. — 1152 p.
4. Attention Is All You Need [Електронний ресурс]. Режим доступу: https://arxiv.org/abs/1706.03762 18.11.2025.
5. Eisenstein J. Introduction to Natural Language Processing. — MIT Press, 2019. — 480 p.
6. GPT-4 Technical Report [Електронний ресурс]. Режим доступу: https://arxiv.org/abs/2303.08774 10.11.2025.
7. Goodfellow I., Bengio Y., Courville A. Deep Learning. — MIT Press, 2016. — 775 p.
8. Language Models are Few-Shot Learners [Електронний ресурс]. Режим доступу: https://arxiv.org/abs/2005.14165 02.12.2025.
9. Vaswani A., Shazeer N., Parmar N., et al. Attention Is All You Need // Neural Information Processing Systems. — 2017.
10. OpenAI API Documentation [Електронний ресурс]. Режим доступу: https://platform.openai.com/docs 04.12.2025.
11. Brown T. B., Mann B., Ryder N., et al. Language Models are Few-Shot Learners // Neural Information Processing Systems. — 2020.
12. BLEU: a Method for Automatic Evaluation of Machine Translation [Електронний ресурс]. Режим доступу: https://aclanthology.org/P02-1040 21.11.2025.
13. Bommasani R., Hudson D., Adeli E., et al. On the Opportunities and Risks of Foundation Models // Stanford University Report. — 2021.

14. GPT-4o Model Overview [Електронний ресурс]. Режим доступу: https://platform.openai.com/docs/models 27.11.2025.
15. Bender E. M., Gebru T., McMillan-Major A., Shmitchell S. On the Dangers of Stochastic Parrots // ACM Conference on Fairness, Accountability, and Transparency. — 2021.
16. ROUGE: A Package for Automatic Evaluation of Summaries [Електронний ресурс]. Режим доступу: https://aclanthology.org/W04-1013 30.11.2025.
17. Hendrycks D., Burns C., et al. Measuring Massive Multitask Language Understanding // International Conference on Learning Representations. — 2021.
18. Evaluating Large Language Models Trained on Code (HumanEval) [Електронний ресурс]. Режим доступу: https://arxiv.org/abs/2107.03374 15.11.2025.
19. Zhao W. X., Zhou K., et al. A Survey of Large Language Models // ACM Computing Surveys. — 2023.
20. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? [Електронний ресурс]. Режим доступу: https://arxiv.org/abs/2310.06770 01.12.2025.
21. Cobbe K., Kaplan J., et al. Training Verifiers to Solve Math Word Problems // Neural Information Processing Systems. — 2021.
22. Measuring Massive Multitask Language Understanding (MMLU) [Електронний ресурс]. Режим доступу: https://arxiv.org/abs/2009.03300 24.11.2025.
23. Liang P., et al. Holistic Evaluation of Language Models // Stanford HELM Report. — 2022.
24. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models [Електронний ресурс]. Режим доступу: https://arxiv.org/abs/2201.11903 12.12.2025.

25. Shanmugam P., Madaan A., et al. Reasoning in Large Language Models // Springer AI Review. — 2023.
26. BERTScore: Evaluating Text Generation with BERT [Електронний ресурс]. Режим доступу: https://arxiv.org/abs/1904.09675 08.12.2025.
27. Chen M., Tworek J., Jun H., et al. Evaluating Large Language Models Trained on Code // OpenAI Research. — 2021.
28. Big-Bench: Beyond the Imitation Game Benchmark [Електронний ресурс]. Режим доступу: https://github.com/google/BIG-bench 07.12.2025.
29. Wei J., Wang X., et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models // Neural Information Processing Systems. — 2022.
30. OpenAI Pricing and Token Usage Guide [Електронний ресурс]. Режим доступу: https://openai.com/pricing 05.12.2025.