

Харківський національний університет радіоелектроніки

Факультет _____ Комп'ютерних наук _____

Кафедра _____ Штучного інтелекту _____

Рівень вищої освіти _____ другий (магістерський) _____

Спеціальність _____ 122 Комп'ютерні науки _____
(код і повна назва)

Тип програми _____ освітньо-наукова _____
(освітньо-професійна або освітньо-наукова)

Освітня програма _____ Системи штучного інтелекту _____
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«_____» _____ 20 ____ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві _____ Норцю Денису Андрійовичу _____
(прізвище, ім'я, по батькові)

1. Тема роботи _____ Дослідження великих мовних моделей для завдань із відповідями на питання _____

затверджена наказом університету від 21 квітня 2025 р. № 295Ст

2. Термін подання студентом роботи до екзаменаційної комісії 6 червня 2025 р.

3. Вихідні дані до роботи Розробити систему дослідження великих мовних моделей для задач питання-відповідь. Система повинна забезпечити запуск обраних мовних моделей, подання вхідних даних, автоматичне формування промптів, збирання відповідей, а також їх оцінювання за допомогою сучасних метрик якості. Очікується реалізація скриптів для попередньої підготовки даних, генерації відповідей, оцінювання результатів, збереження їх у форматі *.json та *.csv, а також побудови візуалізацій на основі оцінок. Перелік використаних програмних засобів: Python 3.x, Visual Studio Code, бібліотеки nltk, rouge-score, bert-score, matplotlib, pandas, OpenAI API, HuggingFace Transformers, Jupyter Notebook.

4. Перелік питань, що потрібно опрацювати в роботі _____

1) Аналіз особливостей застосування великих мовних моделей у сфері питань і відповідей _____

2) Огляд існуючих архітектур моделей та вибір представників для дослідження _____

3) Автоматичне оцінювання якості відповідей із застосуванням сучасних метрик _____

4) Порівняльний аналіз ефективності моделей _____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	21.04.2025	виконано
2	Аналіз предметної галузі	25.04.2025	виконано
3	Актуальність використання та дослідження ВММ	27.04.2025	виконано
4	Визначення нормативних документів	29.04.2025	виконано
5	Вибір великих мовних моделей для дослідження	01.05.2025	виконано
6	Вибір метрик порівняльного аналізу	02.05.2025	виконано
7	Постановка задачі	03.05.2025	виконано
8	Побудова експериментального дослідження	06.05.2025	виконано
9	Вибір та обґрунтування тестових наборів даних	08.05.2025	виконано
10	Формалізація структури експерименту	10.05.2025	виконано
11	Побудова схеми оцінювання результатів	12.05.2025	виконано
12	Запуск моделі BLOOM	15.05.2025	виконано
13	Запуск моделі LLaMA	19.05.2025	виконано
14	Запуск моделі GPT	22.05.2025	виконано
15	Створення скрипту оцінювання відповідей	25.05.2025	виконано
16	Результати оцінювання відповідей	27.05.2025	виконано
17	Опис та аналіз результатів оцінювання	28.05.2025	виконано
18	Написання висновків	30.05.2025	виконано
19	Завершення оформлення, підготовка до захисту	31.05.2025	виконано
20	Захист перед ЕК	06.06.2025	виконано

Дата видачі завдання 21 квітня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____
(підпис)

доц. Олексій Турута
(посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 119 с., 23 рис., 1 табл., 1 дод., 11 джерел.

ВЕЛИКІ МОВНІ МОДЕЛІ, ЕВОЛЮЦІЯ ВІДПОВІДІ, ПИТАННЯ-ВІДПОВІДЬ, BERTSCORE, BLEU, BLOOM, FINE-TUNING, GPT, LLAMA, METEOR, MOVERSCORE, NLP, ROUGE.

Об'єктом дослідження є великі мовні моделі, що використовуються для автоматичної генерації відповідей у завданнях типу «питання-відповідь».

Предметом дослідження є якість відповідей, які генерують ці моделі, а також інструменти й метрики, що використовуються для автоматичної оцінки таких відповідей.

Метою роботи є порівняльний аналіз сучасних великих мовних моделей у задачах питання-відповіді з використанням набору автоматичних метрик, а також дослідження можливостей покращення якості відповідей шляхом донавчання однієї з моделей.

У процесі дослідження використовувалися методи порівняльного аналізу, автоматизованої обробки тексту, семантичного зіставлення текстових відповідей, а також метод донавчання моделей на основі спеціалізованого корпусу даних. Для оцінювання відповідей застосовано метрики BLEU, ROUGE, METEOR, BERTScore та MoverScore.

У результаті роботи було обґрунтовано вибір трьох моделей – GPT, LLaMA та BLOOM. Було побудовано експериментальну модель аналізу, яка дозволяє фіксувати якість відповідей за різними критеріями, порівнювати результати до та після fine-tuning, а також оцінювати адекватність сучасних автоматичних метрик у задачах оцінювання відповіді.

ABSTRACT

Master's thesis contains: 119 pp., 23 fig., 1 tabl., 1 ann., 11 references.

ANSWER EVOLUTION, BERTSCORE, BLEU, BLOOM, FINE-TUNING, GPT, LARGE LANGUAGE MODELS, LLAMA, METEOR, MOVERSCORE, NLP, QUESTION-ANSWER, ROUGE.

The object of the study is large language models used for automatic answer generation in question-and-answer tasks.

The subject of the study is the quality of answers generated by these models, as well as the tools and metrics used to automatically evaluate such answers.

The aim of the study is to compare modern large-scale language models in question-and-answer tasks using a set of automatic metrics, and to investigate the possibilities of improving the quality of answers by retraining one of the models.

The study used the methods of comparative analysis, automated text processing, semantic comparison of textual answers, and a method of model retraining based on a specialized data corpus. The BLEU, ROUGE, METEOR, BERTScore, and MoverScore metrics were used to evaluate the answers.

As a result, the choice of three models – GPT, LLaMA, and BLOOM – was justified. An experimental analysis model has been built that allows recording the quality of answers according to various criteria, comparing the results before and after fine-tuning, and assessing the adequacy of modern automatic metrics in answer evaluation tasks.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів.....	8
Вступ.....	9
1 Аналіз предметної галузі.....	11
1.1 Аналіз предметної галузі, яка визначає діяльність великих мовних моделей питання та відповіді.....	11
1.2 Актуальність використання та дослідження великих мовних моделей.....	15
1.3 Нормативні документи, що регулюють використання систем великих мовних моделей.....	17
1.4 Вибір великих мовних моделей для подальшого дослідження.....	19
1.5 Вибір метрик порівняльного аналізу великих мовних моделей.....	23
1.6 Постановка задачі.....	27
2 Проектування еспериментального дослідження.....	31
2.1 Вибір та обґрунтування тестових наборів даних.....	31
2.2 Формалізація структури експерименту.....	35
2.3 Побудова схеми оцінювання результатів.....	44
2.4 Загальна архітектура експериментальної системи.....	51
3 Реалізація експериментального дослідження та аналіз якості відповідей великих мовних моделей.....	54
3.1 Опис реалізації генерації відповідей.....	54
3.1.1 Запуск моделі BLOOM.....	54
3.1.2 Запуск моделі LLaMA.....	67
3.1.3 Запуск моделі GPT.....	79
3.2 Оцінювання відповідей моделей згідно обраних метрик.....	94
3.2.1 Створення скрипту оцінювання відповідей.....	94
3.2.2 Результати оцінювання відповідей.....	101
3.2.3 Опис та аналіз результатів оцінювання.....	107
Висновки.....	115

Перелік джерел посилання	117
Додаток А Відомість кваліфікаційної роботи	119

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AI – Artificial Intelligence – штучний інтелект;

BERTScore – метрика оцінки семантичної схожості відповідей на основі BERT;

BLEU – Bilingual Evaluation Understudy – метрика якості машинного перекладу;

F1-score – середнє гармонійне між точністю та повнотою в оцінці якості;

GPT – Generative Pre-trained Transformer – генеративна попередньо натренована трансформерна модель;

HF – HuggingFace – бібліотека з відкритим кодом для роботи з LLM.

Inference – процес генерації відповіді на основі навченої моделі;

JSON – JavaScript Object Notation – формат зберігання структурованих даних;

LLM – Large Language Model – велика мовна модель;

LLaMA – Large Language Model Meta AI – відкрита велика мовна модель, розроблена Meta;

NLP – Natural Language Processing – обробка природної мови;

Prompt – текст запити, який подається мовній моделі для генерації відповіді;

ROUGE – Recall-Oriented Understudy for Gisting Evaluation – набір метрик для оцінки якості текстових відповідей;

Token – базова одиниця обробки тексту в мовних моделях (слово або частина слова);

Tokenizer – механізм перетворення тексту в послідовність токенів.

ВСТУП

У сучасному цифровому світі питання ефективної взаємодії людини з інформаційними системами стає все більш актуальним. Користувачі очікують не лише швидкого доступу до великої кількості даних, а й отримання чітких, релевантних і змістовних відповідей на поставлені запитання. У зв'язку з цим системи, засновані на технологіях штучного інтелекту, зокрема на великих мовних моделях (Large Language Models, LLM), набули особливого значення. Вони демонструють високу здатність до генерації тексту, інтерпретації запитів, а також участі у складних інформаційних процесах, що раніше були доступні лише людині.

Однією з ключових задач, у якій можна оцінити інтелектуальний потенціал мовної моделі, є завдання питання-відповіді (Q&A). Це завдання не тільки відображає здатність моделі працювати з текстом, а й перевіряє рівень її «розуміння», вміння витягати знання, логічно міркувати та формулювати відповіді у зрозумілій формі.

З огляду на стрімкий розвиток великих мовних моделей, таких як GPT, LLaMA, BLOOM та інших, постає потреба в їх порівняльному аналізі в умовах однакової задачі. Різні архітектури, обсяги навчальних даних, підходи до генерації відповіді та варіанти відкритості коду формують різну поведінку моделей, яка потребує систематичного вивчення. Це дозволяє зрозуміти, які моделі є більш придатними для практичного застосування в системах питання-відповіді, а також які аспекти їхніх відповідей є сильними або, навпаки, проблемними.

Важливо не лише оцінити самі відповіді, але й визначити, як саме ми можемо це зробити. Автоматичне оцінювання за допомогою метрик BLEU, ROUGE, METEOR, BERTScore, MoverScore дозволяє кількісно вимірювати різні властивості відповіді – від текстової схожості до семантичної відповідності. Однак кожна метрика має свої обмеження, і тому дослідження має на меті не тільки порівняти моделі, але й вивчити,

наскільки об'єктивними та адекватними є ці метрики у випадку генерації відповідей мовними моделями.

У цьому контексті актуальним є також вивчення динаміки якості – еволюції моделі в результаті донавчання (fine-tuning). Це дозволяє оцінити, наскільки модель може адаптуватися до конкретного набору даних або стилю відповіді, а також визначити практичну цінність донавчання в контексті підвищення точності відповідей.

Актуальність теми зумовлена швидким впровадженням ШІ-систем у повсякденне життя, де мовні моделі дедалі частіше використовуються як джерела знань і допоміжні інструменти у вирішенні практичних завдань. Якість відповідей, які вони генерують, прямо впливає на довіру користувачів і ефективність взаємодії з системою.

Крім того, в Україні все активніше впроваджується нормативне регулювання у сфері захисту персональних даних, авторського права та етики застосування ШІ, що підкреслює важливість усвідомленого та обґрунтованого підходу до розробки й використання систем на основі мовних моделей. Дослідження таких систем в умовах реального використання є не лише науково цінним, але й суспільно значущим.

Загалом, результати цієї роботи можуть бути використані для покращення існуючих систем питання-відповіді, розробки нових моделей з урахуванням специфіки задач, а також удосконалення підходів до автоматичного оцінювання результатів генерації тексту. Це відкриває можливості для подальших досліджень у сфері NLP, генеративного ШІ та машинного навчання загалом.

Таким чином, робота має як прикладний, так і теоретичний характер. Вона спрямована на поглиблення розуміння того, як працюють сучасні мовні моделі в умовах складних когнітивних завдань, а також на виявлення ключових факторів, що впливають на якість їхніх відповідей.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз предметної галузі, яка визначає діяльність великих мовних моделей питання та відповіді

Мовні моделі є центральним елементом сучасної обробки природної мови (NLP) і відіграють ключову роль у багатьох завданнях, пов'язаних із розумінням, генерацією та аналізом людської мови. Їхня основна мета – передбачення ймовірності появи наступного слова або послідовності слів на основі контексту. Виходячи з цього принципу, моделі вчаться вловлювати статистичні закономірності в мовленні, граматиці, семантиці та навіть прагматиці мови, що дозволяє їм генерувати логічно узгоджені та змістовні тексти.

Перші мовні моделі будувалися на основі ймовірнісних методів, таких як n-грамні моделі. У таких підходах ймовірність наступного слова оцінюється на основі фіксованої кількості попередніх слів. Хоча ці моделі були простими й швидкими в обчисленні, вони страждали від обмеженого контексту, надмірної залежності від частот у корпусі та проблеми «прокляття розмірності», коли кількість можливих комбінацій зростала експоненційно зі збільшенням розміру контексту.

Із розвитком глибинного навчання почали з'являтися нейронні мовні моделі, які змогли подолати низку обмежень n-грамних підходів. Архітектури на основі рекурентних нейронних мереж (RNN), а згодом і Long Short-Term Memory (LSTM), дозволили моделям ефективно працювати з довгими залежностями в тексті. Проте навіть ці підходи не завжди були здатні повноцінно враховувати глобальний контекст речення або документа, що обмежувало їхні можливості в генеративних або аналітичних завданнях.

Прорив у галузі стався з появою трансформерної архітектури, запропонованої в роботі «Attention is All You Need». Ця архітектура

відмовилася від послідовної обробки тексту на користь механізму самоуваги (self-attention), який дозволив моделі паралельно враховувати всі частини вхідної послідовності. Завдяки цьому трансформери стали фундаментом для створення великих мовних моделей (LLM), таких як BERT, GPT, T5 та інших.

Великі мовні моделі (Large Language Models, LLMs) – це трансформерні архітектури, навчені на масштабних корпусах тексту, що містять мільярди слів. Вони здатні не лише передбачати слова, але й виконувати широкий спектр завдань, включаючи переклад, підсумовування, класифікацію, генерацію коду та відповіді на запитання. Їхня багатофункціональність і здатність до узагальнення знань робить їх потужними інструментами у сфері штучного інтелекту.

Основна ідея навчання LLM полягає в тому, що модель оптимізується для передбачення наступного слова (у випадку автоагресивних моделей, таких як GPT) або заповнення пропусків у тексті (у випадку маскованих моделей, таких як BERT). Ці підходи дозволяють моделі вивчати багатопланове представлення мови, яке охоплює не лише синтаксичні, але й семантичні зв'язки між словами.

Завдяки попередньому тренуванню на загальнодоступних корпусах та подальшому донавчанню на специфічних задачах, великі мовні моделі виявилися надзвичайно адаптивними. Наприклад, GPT-моделі можуть бути використані без донавчання (zero-shot), з мінімальними прикладами (few-shot) або після навчання на специфічному наборі даних (fine-tuning), що значно розширює їхні можливості застосування в реальних умовах.

Окрім технічних можливостей, LLM мають цікаві властивості, пов'язані з появою нових «поведінкових» патернів, які не були явно закладені в навчальні дані. Наприклад, вони можуть імпліцитно навчатися арифметичних операцій, логічного міркування або навігації в структурованих даних. Це явище отримало назву «емерджентних властивостей» моделей.

Попри всі переваги, великі мовні моделі мають свої обмеження. Вони можуть бути схильні до генерації неправдивої інформації (hallucinations), упередженостей (bias) або створення текстів, що виглядають правдоподібно, але є некоректними. Також виклики виникають у сфері етичного використання, зокрема щодо прозорості, безпеки, приватності даних та екологічної вартості тренування таких моделей.

Завдання оцінки якості роботи мовних моделей залишається відкритим. Через складність природної мови та багатозначність формулювань, автоматична оцінка результатів, таких як відповіді на запитання, потребує розробки спеціалізованих метрик. Саме тому дослідження того, що саме ми можемо оцінювати, і які аспекти відповіді коректно вловлюються автоматизованими метриками, має ключове значення для подальшого розвитку цієї галузі.

Вивчення мовних моделей, їхньої архітектури, поведінки та здатності до генерації відповідей є не лише технічним завданням, а й важливою складовою розуміння того, як машини можуть працювати з людською мовою. У цьому контексті аналіз відповіді моделей на запитання – один із найцікавіших способів перевірити, наскільки глибоко модель розуміє зміст тексту, та які компоненти цього розуміння можуть бути зафіксовані за допомогою сучасних NLP-метрик.

Моделі питання-відповіді (Question Answering, Q&A) – це підклас систем обробки природної мови, мета яких полягає в тому, щоб надавати точні та змістовні відповіді на задані запитання на основі певного джерела інформації або загальних знань. Вони моделюють одне з найбільш природних способів взаємодії людини з інформацією, дозволяючи не просто шукати документи, як це робить звичайний пошуковий механізм, а одразу отримувати безпосередню відповідь на запит.

Q&A-моделі зазвичай класифікують за типом вхідних даних. Найпоширенішими є моделі, які працюють у рамках читання з

контекстом (reading comprehension), коли разом із запитанням подається текстовий фрагмент (контекст), на основі якого потрібно знайти або згенерувати відповідь. У такому випадку відповідь має бути тісно пов'язана з контекстом, а іноді й дослівно його повторювати. Інший тип – відкриті моделі Q&A, які не мають явного контексту, а формують відповідь на основі знань, набутих під час попереднього навчання або шляхом звернення до зовнішньої бази знань чи пошуку.

Існують також поділи моделей за типом відповіді: екстрактивні моделі витягують фрагмент з контексту, який відповідає на запитання, тоді як генеративні моделі будують відповідь заново, можливо, комбінуючи декілька частин інформації або перефразовуючи її. Екстрактивний підхід був типовим для ранніх моделей на основі BERT, тоді як сучасні LLM, як-от GPT, демонструють сильні результати в генеративному Q&A.

Розвиток моделей питання-відповіді мав особливе значення в академічних бенчмарках, таких як SQuAD, Natural Questions, HotpotQA, які стали стандартом для оцінки моделей. У таких датасетах зазвичай кожне запитання супроводжується правильними відповідями, що дозволяє точно вимірювати продуктивність моделей за метриками точності, повноти та інших.

Q&A-завдання також тісно пов'язані з більш загальним завданням машинного читання та розуміння тексту, тому що для надання правильної відповіді модель має не лише знаходити релевантні фрагменти, а й розуміти зв'язки між словами, логіку викладу та іноді навіть приховані імплікації. Це робить Q&A ідеальним тестовим середовищем для перевірки глибини «розуміння» мовної моделі.

З практичного боку, системи питання-відповіді застосовуються в численних сферах – від чат-ботів та віртуальних помічників до медичних інформаційних систем і пошукових платформ. Висока точність відповідей у таких системах є критичною, оскільки навіть невелика помилка може спричинити хибні висновки або втрату довіри користувача.

Моделі Q&A стають все більш складними завдяки інтеграції знань із різних джерел, багатомовній підтримці та здатності до дедуктивного або причинно-наслідкового міркування. Проте зростає і складність їхньої оцінки – з'являється потреба не лише перевіряти фактологічну правильність, але й аналізувати формулювання, логічну послідовність і повноту відповіді. Саме тому дослідження методів автоматичної оцінки якості відповідей у Q&A-моделях має надзвичайно важливе значення.

1.2 Актуальність використання та дослідження великих мовних моделей

Актуальність дослідження великих мовних моделей у контексті завдань питання-відповіді зумовлена стрімким зростанням обсягів інформації, яку необхідно обробляти, систематизувати та ефективно використовувати. У сучасному цифровому середовищі користувачі очікують не лише швидкого доступу до даних, але й отримання чітких, релевантних відповідей на конкретні запитання. Традиційні методи пошуку або ручного аналізу інформації виявляються повільними, неефективними або надто ресурсоємними, тоді як сучасні мовні моделі здатні автоматизувати цей процес на якісно новому рівні.

Великі мовні моделі, натреновані на об'ємних корпусах тексту, демонструють вражаючі результати в генерації логічно зв'язаного, семантично насиченого та граматично правильного тексту. Їх використання в задачах питання-відповіді дозволяє швидко отримувати точну інформацію навіть у випадках, коли запитання є складним, контекстуальним або двозначним. Це відкриває широкі можливості для застосування в освіті, юриспруденції, охороні здоров'я, наукових дослідженнях та бізнес-аналітиці.

Проте зростання складності моделей породжує нові виклики – оцінка якості їхньої роботи перестає бути тривіальним завданням. У багатьох

випадках відповідь може бути формально правильною, але змістовно неточною, або навпаки – відповідь містить корисну інформацію, але не збігається з очікуваним формулюванням. Це унеможлиблює використання лише традиційних метрик, які оцінюють збіг за словами чи фразами, і вимагає нових підходів до аналізу.

Дослідження NLP-метрик, таких як ROUGE, BLEU, BERTScore, у контексті Q&A дозволяє краще зрозуміти, що саме ми можемо автоматично виміряти в роботі мовної моделі. Це особливо важливо у випадках, коли масштабне ручне оцінювання є неможливим або надто дорогим. Автоматизована оцінка якості відповідей може забезпечити ефективну перевірку моделей на великих обсягах даних, підтримати процес відбору найкращих архітектур та покращити процес fine-tuning.

Важливо й те, що подібні дослідження сприяють підвищенню прозорості в системах ШІ. Якщо ми здатні формально пояснити, чому модель дала правильну або неправильну відповідь, це підвищує рівень довіри користувачів, спрощує налагодження системи та зменшує ризики її помилкового використання. У критичних сферах, як-от медицина чи юриспруденція, така пояснюваність має ключове значення.

Окрім цього, розвиток інструментів оцінки відповіді моделі може стати основою для створення нових систем самокорекції, де модель не лише генерує відповідь, але й «оцінює» свою впевненість у ній. Це може покращити якість інтерактивних систем, таких як чат-боти чи віртуальні помічники, які взаємодіють із користувачами в режимі реального часу.

Таким чином, тема є надзвичайно актуальною як з теоретичної, так і з прикладної точки зору. Вона дозволяє не лише поглибити розуміння внутрішніх механізмів роботи великих мовних моделей, а й закласти фундамент для практичного підвищення їхньої точності, надійності та відповідності реальним запитам користувачів.

1.3 Нормативні документи, що регулюють використання систем великих мовних моделей

Використання систем, заснованих на великих мовних моделях для завдань питання-відповіді, в Україні регулюється низкою нормативно-правових актів, які охоплюють сфери інтелектуальної власності, персональних даних, захисту прав споживачів та електронних довірчих послуг. Нижче наведено основні закони та їхній вплив на діяльність таких систем:

– Закон України «Про авторське право і суміжні права» від 1 грудня 2022 року № 2811-IX. Згідно зі статтею 33, такі об'єкти охороняються правом особливого роду (*sui generis*), а суб'єктами цього права є особи, які мають майнові права або ліцензійні повноваження щодо відповідної комп'ютерної програми. Це означає, що результати, створені системами на основі великих мовних моделей, можуть підпадати під цей захист, і права на них належать власникам або ліцензіатам відповідного програмного забезпечення [3];

– Закон України «Про захист персональних даних» № 2297-VI від 1 червня 2010 року. Цей закон регулює обробку персональних даних, що є критично важливим при використанні мовних моделей, які можуть обробляти або генерувати інформацію, що містить персональні дані. Системи питання-відповіді повинні забезпечувати дотримання принципів законності, пропорційності та цільового призначення при обробці таких даних, а також гарантувати права суб'єктів персональних даних [4];

– Закон України «Про захист прав споживачів» № 1023-XII від 12 травня 1991 року. У випадку, коли системи на основі мовних моделей надають інформаційні послуги споживачам, вони підпадають під дію цього закону. Це зобов'язує забезпечувати належну якість послуг, надавати достовірну інформацію про їхні характеристики та гарантувати права споживачів на безпеку та захист від недобросовісної практики [5];

– Закон України «Про електронні довірчі послуги» № 2155-VIII від 5 жовтня 2017 року. Якщо системи питання-відповіді використовуються для надання електронних послуг, що потребують ідентифікації або автентифікації користувачів, вони повинні відповідати вимогам цього закону щодо електронних підписів, печаток та інших довірчих послуг [6];

– концепція розвитку штучного інтелекту в Україні, схвалена розпорядженням Кабінету Міністрів України № 1556-р від 2 грудня 2020 року. Цей документ визначає пріоритети та напрями розвитку технологій штучного інтелекту, включаючи питання етики, правового регулювання та інтеграції ШІ в різні сфери суспільного життя. Він підкреслює важливість створення відповідної нормативно-правової бази для використання ШІ, що безпосередньо стосується і систем питання-відповіді [11].

Окрім зазначених законів, варто враховувати міжнародні стандарти та рекомендації щодо відповідального використання штучного інтелекту. Наприклад, «Рекомендації щодо відповідального використання ШІ», розроблені Національним інститутом інтелектуальної власності України, надають керівні принципи для розробників та користувачів ШІ-систем, спрямовані на забезпечення етичності, прозорості та дотримання прав людини.

Важливо також враховувати аспекти академічної доброчесності при використанні ШІ в освітньому процесі. Зокрема, стаття «Академічна доброчесність та штучний інтелект» аналізує проблеми дотримання академічної доброчесності в умовах використання ШІ та пропонує заходи для забезпечення чесності та прозорості в освітньому середовищі.

Крім того, використання ШІ в правоохоронній діяльності також потребує врахування правових аспектів. Стаття «Використання ChatGPT при виявленні та розслідуванні кримінальних правопорушень» обговорює можливості та обмеження застосування мовних моделей у кримінальному процесі, підкреслюючи необхідність дотримання законодавства та етичних норм.

Таким чином, діяльність систем, що базуються на великих мовних моделях для завдань питання-відповіді, в Україні регулюється комплексом законів та нормативних актів, які охоплюють різні аспекти: від інтелектуальної власності та захисту персональних даних до прав споживачів та електронних послуг. Розробники та користувачі таких систем повинні ретельно дотримуватися цих нормативних вимог, забезпечуючи законність, етичність та безпеку їхнього використання.

1.4 Вибір великих мовних моделей для подальшого дослідження

У межах дослідження завдань питання-відповіді доцільно розглянути декілька популярних представників великих мовних моделей, які показали високу ефективність у генерації відповідей, обробці контексту та демонструють емерджентну поведінку. Нижче наведено опис шести актуальних моделей, що репрезентують різні архітектури та підходи до вирішення задачі Q&A.

GPT-4 (Generative Pre-trained Transformer) – це автоагресивна мовна модель, розроблена компанією OpenAI. Вона є продовженням лінійки GPT-моделей, основна особливість якої полягає в генерації тексту на основі передбачення наступного токена. GPT-4 володіє надзвичайно широкими знаннями, отриманими під час попереднього навчання на масштабних відкритих корпусах, і демонструє високі результати в завданнях відкритого типу питання-відповіді, завдяки гнучкості, здатності до дедуктивного мислення та генерації складних, багаторівневих відповідей.

LLaMA (Large Language Model Meta AI) – це серія мовних моделей, розроблена компанією Meta. LLaMA орієнтована на ефективне навчання з відкритими даними, при цьому зберігаючи високу продуктивність навіть при меншій кількості параметрів у порівнянні з аналогами. Моделі LLaMA є зручними для дослідників завдяки відкритому коду та можливості

тонкого налаштування під конкретні задачі. У Q&A-сценаріях LLaMA продемонструвала здатність формулювати точні, контекстно релевантні відповіді та добре працює як у генеративних, так і в екстрактивних режимах.

BLOOM (BigScience Large Open-science Open-access Multilingual Language Model) – це багатомовна велика мовна модель, розроблена в рамках відкритого наукового колаборативного проєкту BigScience. Її архітектура базується на трансформерах, а тренування проводилося на великій кількості текстів більш ніж 40 мовами. BLOOM показує гідні результати в задачах питання-відповіді, особливо у випадках, де потрібна мовна різноманітність або робота з нестандартними запитамі. Основною перевагою є повна відкритість моделі, що дозволяє дослідникам вивчати її на глибшому рівні.

BERT (Bidirectional Encoder Representations from Transformers) – це одна з перших моделей, яка застосувала двонаправлений підхід до обробки тексту на основі трансформерів. Хоча BERT не є генеративною моделлю в класичному сенсі, вона показала видатні результати в екстрактивних Q&A-завданнях, таких як SQuAD. BERT не генерує відповідь повністю, а натомість виділяє відповідний фрагмент з контексту. Вона стала базовою архітектурою для великої кількості спеціалізованих моделей у вузьких доменах (наприклад, BioBERT, LegalBERT).

T5 (Text-To-Text Transfer Transformer) – це модель, запропонована Google Research, яка уніфікує всі NLP-завдання у форматі «текст-в-текст». Це означає, що як вхід, так і вихід подається у вигляді рядків, незалежно від задачі. У випадку Q&A модель отримує запитання і контекст як єдиний текстовий вхід і генерує текстову відповідь. T5 відзначається гнучкістю, високою якістю узагальнення і хорошими результатами на широкому спектрі завдань, зокрема і на складних багатоконтекстних запитаннях.

FLAN-T5 (Fine-tuned LAnguage Net) – це вдосконалена версія T5, яка була додатково навчена на задачах інструкційного типу. Вона демонструє

ще вищу продуктивність у Q&A-завданнях, особливо у few-shot або zero-shot сценаріях. FLAN-T5 здатна краще адаптуватися до нових типів запитань завдяки більш універсальним навчальним шаблонам.

Таким чином, кожна з цих моделей має свої сильні сторони та підходить до вирішення Q&A-завдань із різних позицій – від екстрактивних до генеративних стратегій, від одно- до багатомовної підтримки, від закритих до повністю відкритих рішень. У наступному підпункті можна обґрунтувати вибір трьох моделей для подальшого порівняльного аналізу.

З-поміж широкого спектру існуючих великих мовних моделей, доцільно зосередити увагу на тих, які демонструють високі результати в завданнях питання-відповіді, мають різне походження та архітектурні особливості, а також представляють різні підходи до відкритості й доступності моделей. Такий вибір дозволяє провести об'єктивне порівняння моделей з різними філософіями розробки, що, у свою чергу, забезпечує більш повне розуміння їхньої поведінки та ефективності в задачах Q&A.

GPT (зокрема GPT-3.5 або GPT-4) є репрезентативною моделлю закритого комерційного типу, яка вирізняється високою точністю відповідей, розумінням складного контексту та здатністю до генеративного мислення. Ця модель використовується як основа для великої кількості інтерактивних систем, таких як чат-боти та аналітичні платформи, що свідчить про її практичну ефективність у реальних умовах. Її включення до дослідження дозволяє розглядати state-of-the-art рівень генерації відповідей у задачах питання-відповіді.

LLaMA, навпаки, є прикладом відкритої моделі, орієнтованої на дослідницьке використання. Вона пропонує хорошу продуктивність навіть при меншій кількості параметрів, ніж GPT, що робить її привабливою для адаптації під конкретні задачі. У порівнянні з комерційними моделями, LLaMA дозволяє більш гнучко контролювати навчальний процес,

застосовувати донавчання або аналізувати внутрішні механізми прийняття рішень.

BLOOM, своєю чергою, є унікальним прикладом відкритої, багатомовної великої мовної моделі, створеної в рамках наукового колективу BigScience. Вона дозволяє дослідити, як багатомовність та спільна розробка впливають на якість відповідей. Завдяки своїй архітектурі та відкритості, BLOOM надає можливість більш глибокого аналізу результатів, включаючи міжмовну точність, генеративні властивості та обробку нестандартних запитів.

Таким чином, вибір моделей GPT, LLaMA та BLOOM забезпечує репрезентативність дослідження як у технічному, так і в концептуальному вимірі. Це дозволяє порівняти можливості комерційних і відкритих рішень, зіставити генеративні властивості моделей, а також оцінити їхню здатність давати точні, послідовні та релевантні відповіді в межах завдання питання-відповіді.

Такий підхід не лише забезпечує різноманітність у розглянутих архітектурах, а й дає змогу сформулювати більш обґрунтовані висновки щодо ефективності моделей, особливостей їхніх відповідей і ролі автоматичних метрик у визначенні їхньої якості.

Обрані для дослідження моделі – GPT, LLaMA та BLOOM – репрезентують різні підходи до побудови великих мовних систем і дозволяють комплексно оцінити можливості сучасних архітектур у завданнях питання-відповіді. Кожна з них має унікальні характеристики, які роблять її цінною для дослідницького аналізу: GPT як потужна комерційна модель, LLaMA як гнучке дослідницьке рішення з відкритим кодом, та BLOOM як приклад багатомовної моделі, створеної у форматі відкритої науки.

GPT вирізняється високою якістю генерації, здатністю працювати з неструктурованими запитам та демонструє сильні результати навіть у zero-shot сценаріях. Її участь у дослідженні дозволяє встановити верхню

межу якості, на яку орієнтуються інші моделі. Однак її закритість та обмежений контроль над параметрами створюють певні бар'єри для глибшого аналізу або кастомізації.

LLaMA, навпаки, є зручною платформою для гнучкого налаштування та експериментів, що особливо важливо у контексті дослідження *fine-tuning* і впливу додаткового навчання на якість відповідей. Її відкритість дозволяє детально аналізувати внутрішні механізми прийняття рішень, а менша кількість параметрів робить її більш ефективною в ресурсних обмеженнях, зберігаючи при цьому прийнятну якість результатів.

BLOOM додає до порівняння багатомовний аспект, а також демонструє, яким чином відкриті, колаборативні підходи до навчання мовних моделей можуть впливати на їхню продуктивність. Особливо цікаво оцінити, як BLOOM поводить себе в різних мовних і тематичних контекстах, та наскільки добре справляється з відповідями, що потребують генеративного мислення.

Загалом, обрані моделі утворюють збалансовану сукупність рішень, які разом дозволяють провести всебічний аналіз якості відповідей у Q&A-завданнях. Це дає змогу оцінити не лише абсолютні результати, а й зрозуміти сильні та слабкі сторони кожної архітектури, а також сформулювати рекомендації щодо вибору моделі залежно від конкретних вимог та обмежень задачі.

1.5 Вибір метрик порівняльного аналізу великих мовних моделей

Для повноцінного порівняльного аналізу якості відповідей великих мовних моделей у завданні питання-відповіді необхідно обрати відповідний набір метрик, які дозволяють кількісно оцінити різні аспекти відповіді: від точності передачі інформації до семантичної відповідності та стилістичної адекватності. Оскільки в задачі Q&A може бути кілька

правильних варіантів відповіді, метрики мають враховувати не лише буквальный збіг з еталонною відповіддю, але й її смислову близькість.

Однією з найпоширеніших метрик у сфері оцінювання генерації тексту є BLEU (Bilingual Evaluation Understudy). Вона була вперше запроваджена для оцінки машинного перекладу і базується на підрахунку кількості n-грам у відповіді моделі, які збігаються з n-грамами в еталонному тексті. BLEU добре підходить для коротких, точно сформульованих відповідей, але має обмеження в задачах, де важлива смислова варіативність: навіть правильна відповідь, сформульована по-іншому, може отримати низьку оцінку.

Ще одна важлива метрика – ROUGE (Recall-Oriented Understudy for Gisting Evaluation), яка використовується переважно для оцінювання підсумовування, але також ефективна в задачах Q&A. ROUGE вимірює перекриття між відповіддю моделі та еталонною відповіддю на рівні слів або фраз, з акцентом на повноту (recall). Вона краще, ніж BLEU, фіксує ступінь охоплення ключової інформації, проте також залишається чутливою до формулювання.

Окремо варто виділити BERTScore, який є метрикою нового покоління. На відміну від BLEU та ROUGE, що базуються на поверхневому аналізі тексту, BERTScore використовує попередньо натреновану трансформерну модель для обчислення подібності між векторними представленнями слів у відповіді та еталоні. Це дозволяє метриці враховувати семантичну подібність і бути менш чутливою до синтаксичних відмінностей. BERTScore надає змогу оцінювати не лише точність, а й глибину розуміння моделі.

У задачах, де одна відповідь може мати кілька варіантів формулювання, важливою є METEOR – метрика, яка враховує синонімію, перестановки слів та морфологічні варіації. Вона забезпечує більш «гуманний» підхід до оцінювання, оскільки наближена до принципів людського сприйняття якості тексту. METEOR особливо корисна для

аналізу генеративних моделей, які не просто копіюють, а вільно формулюють відповіді.

У контексті семантичного аналізу все більшої популярності набуває метрика MoverScore, яка, подібно до BERTScore, використовує векторні представлення слів, але додатково застосовує принцип оптимального транспорту для оцінювання схожості між текстами. Цей підхід дозволяє не лише оцінити наявність релевантної інформації, а й її структурне розташування, що важливо в задачах, де відповідь має логічну послідовність.

Окрім автоматичних метрик, іноді доцільно враховувати людське оцінювання (human evaluation), яке, хоч і є затратним, все ще вважається «золотим стандартом». Проте в рамках автоматизованого дослідження така оцінка буде застосовуватися радше як допоміжна або контрольна.

Для цілей даного дослідження доцільно зосередитися на метриках, які репрезентують різні підходи до аналізу: BLEU і ROUGE як класичні статистичні метрики, BERTScore як семантично орієнтована, METEOR як метрика з урахуванням синонімії та варіативності, а також MoverScore як приклад більш глибокого векторного аналізу.

Крім того, різні метрики по-різному реагують на помилки моделі. Наприклад, BLEU суворо карає за відхилення від шаблону, тоді як BERTScore може розпізнати відповідь як релевантну навіть при зміні структури. Це дозволить оцінити не лише абсолютну якість відповіді, а й характер поведінки кожної моделі: наскільки вона схильна до шаблонності, перефразування чи генерації нового змісту.

Інтеграція декількох метрик у межах одного дослідження створює умови для багатовимірного аналізу. Наприклад, висока оцінка за ROUGE при низькому BERTScore може свідчити про поверхневий збіг, але недостатню смислову точність. Навпаки, висока семантична подібність при низькому збігові за словами може вказувати на добру генеративну якість, але вільний стиль формулювання.

Таким чином, вибір метрик не є формальністю, а критично важливим кроком для забезпечення об'єктивного та глибокого аналізу відповідей мовних моделей. Застосування декількох підходів дозволяє уникнути упередженості окремих метрик і забезпечує ширше уявлення про сильні й слабкі сторони кожної з моделей.

У межах даної роботи буде використано п'ять основних метрик: BLEU, ROUGE, BERTScore, METEOR та MoverScore. Такий вибір дозволяє поєднати класичні та сучасні підходи до автоматичного оцінювання, забезпечуючи як формальну точність, так і семантичну глибину аналізу. Кожна з метрик буде застосовуватися до відповідей трьох моделей (GPT, LLaMA, BLOOM) з подальшим порівнянням отриманих результатів.

Застосування вищезазначених метрик дозволить не лише встановити, яка з моделей демонструє найвищі показники за окремими критеріями, але й відповісти на ключове питання дослідження: які саме аспекти відповіді може оцінити машина, а що залишається за межами автоматичного аналізу.

Проведений аналіз метрик для оцінювання якості відповідей великих мовних моделей дозволяє зробити висновок про необхідність комплексного підходу до автоматичного вимірювання результатів. Жодна з метрик не є універсальною або достатньою для повного опису якості відповіді, оскільки різні аспекти, як-от точність формулювання, семантична відповідність, стилістична гнучкість чи логічна послідовність, вимагають різних методів аналізу.

Класичні метрики, такі як BLEU та ROUGE, забезпечують базову перевірку збігу текстових одиниць, проте демонструють обмежену ефективність у завданнях, де можлива варіативність формулювання правильної відповіді. Їхня точність особливо падає в умовах генеративних моделей, де відповідь, хоч і правильна, може суттєво відрізнятися за формою від еталону.

Семантично орієнтовані метрики – зокрема BERTScore, METEOR та MoverScore – надають можливість враховувати глибші рівні відповідності, зокрема синонімію, семантичну близькість і логічну зв'язність. Вони є критично важливими для аналізу сучасних LLM, які не просто копіюють інформацію, а синтезують її, створюючи нові, але релевантні формулювання відповідей.

Таким чином, об'єднання кількох типів метрик дозволяє створити збалансовану систему оцінки, що враховує як формальні, так і змістовні характеристики відповіді. Це, у свою чергу, забезпечує глибше розуміння як самої якості відповідей, так і особливостей поведінки моделей у контексті завдання питання-відповіді.

Вибір зазначених метрик створює надійне підґрунтя для подальшого порівняльного аналізу моделей GPT, LLaMA та BLOOM. Це дозволить не лише зафіксувати відмінності у їхній продуктивності, а й відповісти на ширше дослідницьке питання: які характеристики відповіді ми можемо ефективно виміряти автоматично, а які залишаються в межах людського судження.

1.6 Постановка задачі

Постановка задачі в даному дослідженні полягає у всебічному аналізі якості відповідей великих мовних моделей у задачах типу «питання-відповідь» (Q&A), з метою виявлення сильних та слабких сторін моделей, а також меж застосовності сучасних автоматичних метрик оцінки. У межах цього завдання необхідно дослідити поведінку трьох обраних моделей – GPT, LLaMA та BLOOM – при генерації відповідей на типові запитання із заданим контекстом, а також у сценаріях відкритого типу, коли контекст не задається явно.

Перше завдання полягає у формуванні контрольованого набору запитань і відповідей. Цей набір повинен охоплювати різні типи питань:

фактичні (наприклад, «Хто відкрив Америку?»), логічні («Чому сталася Друга світова війна?»), обчислювальні («Скільки буде 15% від 200?»), а також запитання, що потребують узагальнення («Який зміст цього тексту?»). Це дозволить протестувати моделі на здатність працювати з різними когнітивними рівнями складності.

Другим етапом є запуск кожної з обраних моделей на цьому наборі запитань. Відповіді кожної моделі фіксуються для подальшого аналізу. У випадку GPT використовується API або локальний інтерфейс із відповідною версією моделі (наприклад, GPT-3.5 або GPT-4), у випадку LLaMA та BLOOM – відповідні відкриті реалізації, доступні через HuggingFace або інші платформи. Усі моделі мають бути налаштовані в однакових умовах (однакові промпти, темплейти, параметри генерації), щоб забезпечити коректність порівняння.

Далі виконується оцінка кожної відповіді за допомогою набору автоматичних метрик: BLEU, ROUGE, METEOR, BERTScore і MoverScore. Кожна з цих метрик надає незалежну кількісну оцінку якості відповіді. Наприклад, BLEU та ROUGE дозволяють оцінити ступінь перекриття з еталонною відповіддю, а BERTScore і MoverScore – семантичну подібність. Оцінка здійснюється в автоматичному режимі, а результати заносяться до таблиці для подальшого аналізу.

Особливу увагу буде приділено виявленню випадків, коли метрики дають суперечливі оцінки. Наприклад, відповідь може мати низький BLEU, але високий BERTScore – це сигналізує про відмінність у формулюванні, але збереження змісту. Такі випадки мають бути ідентифіковані й розібрані вручну для виявлення слабких місць кожної метрики.

Наступним етапом є побудова порівняльного аналізу між трьома моделями. За кожною метрикою виводиться середнє значення по всіх відповідях. Також можна застосувати візуалізацію результатів – наприклад, за допомогою гістограм або boxplot-графіків, щоб побачити розподіл

оцінок. Це дозволить визначити, яка з моделей показує найвищу стабільність і якість за кожним критерієм.

Окремим етапом дослідження буде аналіз еволюції якості відповідей. Для цього буде обрана одна з моделей (наприклад, LLaMA) і проведено її донавчання (fine-tuning) на підмножині запитань із відповідями. Мета – оцінити, чи дійсно якість відповідей покращиться після адаптації моделі до конкретної тематики або стилю подачі.

Fine-tuning буде виконано з використанням відкритих фреймворків (наприклад, PEFT або LoRA), що дозволяють ефективно оновлювати модель навіть при обмежених ресурсах. Після донавчання модель буде повторно протестована на тому ж наборі питань, і результати будуть знову проаналізовані за тими самими метриками. Порівняння до та після fine-tuning дозволить оцінити вплив адаптації на якість.

Також буде проведено якісний аналіз типових помилок моделей. Для цього обираються приклади, де оцінки метрик були низькими, і вручну аналізується, чи дійсно відповідь була помилковою, або ж метрика не змогла адекватно її оцінити. Це дасть змогу краще зрозуміти обмеження кожної метрики та виявити «сліпі зони» в автоматичному оцінюванні.

Окрему увагу буде приділено визначенню меж застосовності кожної моделі. Наприклад, одна з моделей може краще працювати в коротких запитаннях із чітким контекстом, тоді як інша буде сильнішою в генеративних, відкритих запитаннях. Це дозволить сформулювати рекомендації щодо вибору моделі залежно від типу задачі.

Під час виконання дослідження буде сформована узагальнена таблиця продуктивності, де моделі будуть ранжовані за сукупними оцінками по всіх метриках. При цьому буде враховано як середні значення, так і дисперсію результатів, щоб визначити не лише «найкращу» модель, але й найбільш стабільну.

Результати будуть інтерпретовані в контексті практичного використання систем питання-відповіді: наскільки автоматична оцінка

може замінити людську, які аспекти відповіді залишаються поза межами автоматичного аналізу, та що можна покращити в системах оцінювання для майбутніх досліджень.

Таким чином, постановка задачі включає: вибір моделей і метрик, формування датасету запитань, генерацію відповідей, автоматичну та частково ручну оцінку, порівняння моделей між собою, проведення fine-tuning однієї з моделей та повторну оцінку, а також аналіз зміни якості. Усі ці етапи спрямовані на досягнення головної мети дослідження – визначення того, що саме ми можемо об'єктивно виміряти у відповіді великих мовних моделей, а що залишається суб'єктивним або складним для автоматичної інтерпретації.

2 ПРОЄКТУВАННЯ ЕСПЕРЕМЕНТАЛЬНОГО ДОСЛІДЖЕННЯ

2.1 Вибір та обґрунтування тестових наборів даних

У процесі оцінювання якості великих мовних моделей у задачах питання-відповіді критичне значення має вибір тестових наборів даних. Саме від характеристик обраного корпусу залежить об'єктивність і достовірність висновків щодо продуктивності моделей. Тестові датасети виступають джерелом як запитань, так і еталонних відповідей, з якими порівнюються результати генерації моделей. Тому до їх вибору слід підходити з урахуванням типів запитань, структури відповідей, мови, а також тематичного охоплення.

Існує низка загальноживаних датасетів, які визнані в науковій спільноті як стандарт для оцінювання систем Q&A. Їх особливістю є наявність великої кількості перевірених запитань з відповідями, що дозволяє забезпечити стабільне середовище для тестування. Крім того, багато з таких датасетів були використані при попередньому навчанні моделей, що дає змогу оцінити їхню здатність до відтворення чи генерації відповідей за вже відомими зразками.

Разом з тим важливо брати до уваги тип завдання, що вирішується моделлю. Наприклад, екстрактивні моделі краще працюють на датасетах, де відповідь буквально присутня в тексті, тоді як генеративні моделі можуть створювати нові формулювання, що також слід враховувати при виборі метрик. Деякі датасети надають лише запитання й контекст, інші – дозволяють вільну генерацію без контексту. Відповідно, різні типи датасетів дозволяють перевірити різні аспекти поведінки моделі.

Для забезпечення коректності порівняння в межах даної роботи використовуються виключно англомовні датасети. Такий вибір зумовлений тим, що більшість великих мовних моделей, які беруть участь у дослідженні, були попередньо навчені саме на англомовних корпусах і

демонструють найвищу стабільність та якість генерації саме в англомовному середовищі.

Таким чином, вибір тестових датасетів є важливим етапом проєктування експерименту, оскільки саме вони формують основу для порівняння моделей та аналізу їхньої поведінки в умовах реальних задач. У наступних пунктах буде обґрунтовано, які саме набори даних було обрано для проведення тестування, а також розглянуто їхню структуру, специфіку й відповідність поставленій меті дослідження.

Для проведення дослідження якості відповідей великих мовних моделей було розглянуто низку англомовних датасетів, які традиційно використовуються для задач типу питання-відповіді. Їхній вибір обумовлений різноманітністю форматів, типів запитань, обсягом контекстної інформації та ступенем складності. Вивчення кількох альтернативних джерел дозволяє сформувати збалансовану базу для тестування та забезпечити репрезентативність експерименту.

Кожен із розглянутих датасетів має власну специфіку – одні орієнтовані на короткі прямі відповіді, інші передбачають складні логічні ланцюги та обґрунтування. Такий підхід дає змогу виявити, які моделі краще працюють із фактологічними запитаннями, а які – з багатоступеневими міркуваннями:

- *soqa*. діалоговий датасет, у якому питання й відповіді подаються у формі послідовної бесіди між людиною та моделлю, з урахуванням попередніх відповідей і контексту;

- *gase*. набір запитань на основі текстів, орієнтованих на шкільні тести читання, розділений на рівні складності (середня школа та старша школа), включає як фактологічні, так і аналітичні запитання;

- *triviaqa*. збірка відкритих запитань, часто сформульованих складно й неочевидно, з підтримкою великого контексту, що дозволяє оцінити здатність моделі знаходити інформацію в довгих документах;

– hotpotqa. датасет, який вимагає багатокрокового міркування для відповіді, містить запитання, що вимагають комбінування кількох фрагментів інформації з різних частин тексту;

– natural questions. колекція реальних запитань користувачів Google з відповідями, витягнутими з Вікіпедії, включає як короткі, так і довгі варіанти відповідей, часто з неоднозначним формулюванням запити.

Аналіз зазначених наборів даних дозволив визначити найпридатніші для експерименту джерела. Датасети охоплюють як діалогові, так і енциклопедичні сценарії використання, мають достатній обсяг прикладів, і розміщені у форматі, зручному для обробки великими мовними моделями.

Після аналізу кількох потенційних датасетів для тестування було прийнято рішення зосередити увагу на наборі даних Natural Questions, який розроблений командою Google Research. Цей датасет відрізняється своєю реалістичністю, обсягом, складністю та високим ступенем відповідності завданню, яке ставиться в межах цієї дипломної роботи – порівняння якості відповідей великих мовних моделей на природні запитання користувачів.

Однією з головних причин вибору Natural Questions є те, що запитання в цьому датасеті взяті з реальних пошукових запитів, які користувачі вводили в Google. Це дозволяє максимально наблизити експериментальні умови до реальних сценаріїв використання мовних моделей. На відміну від деяких інших датасетів, де запитання сформульовані штучно або навмисно ускладнені, Natural Questions представляє автентичні формулювання, які можуть містити двозначності, помилки, нестандартну структуру – тобто ті особливості, з якими система стикається в реальній роботі.

Ще однією перевагою Natural Questions є наявність як коротких, так і довгих відповідей. Коротка відповідь зазвичай складається з кількох слів і виконує роль швидкого факту, тоді як довга відповідь – це розширений текстовий фрагмент з Вікіпедії, який надає ширший контекст. Це дозволяє

протестувати, наскільки модель здатна адаптувати відповідь до формату, а також наскільки вона вміє вибудовувати логічно завершені, інформативні фрагменти тексту.

Структура Natural Questions добре підходить для генеративного підходу до відповіді, оскільки модель має справу не лише з простими витягами, а й із повноцінними мовними завданнями. Завдяки цьому зростає актуальність застосування метрик BERTScore, METEOR, MoverScore, які оцінюють не просто формальний збіг, а й семантичну насиченість відповіді. Крім того, наявність різних рівнів складності дає змогу дослідити, як моделі поведуться у простих і складних випадках.

Датасет Natural Questions також має велику кількість прикладів, що забезпечує статистичну надійність експериментів. Можливість відбору підмножини запитань за тематикою або структурою дозволяє формувати контрольовані підвибірki для fine-tuning, що особливо важливо для подальшого навчання моделей на специфічних випадках.

Ще одним важливим аспектом є наявність супровідної розмітки у форматі JSON із чітко визначеними полями, що значно спрощує автоматичну обробку, підготовку до завантаження в модель і подальше порівняння відповідей. Це дозволяє уникнути витрат часу на попередню очистку даних, що є перевагою у рамках обмеженого часу дипломної роботи.

Варто також зазначити, що цей датасет є відкритим і широко використовуваним у наукових дослідженнях. Його підтримка від авторитетної команди Google додає впевненості в якості даних. Наявність попередніх досліджень на основі Natural Questions дає змогу порівнювати результати та обирати обґрунтовані методики обробки.

Таким чином, Natural Questions забезпечує збалансовану, гнучку та репрезентативну основу для тестування великих мовних моделей у завданнях питання-відповіді. Його використання дозволяє реалізувати

експеримент, який буде як методологічно обґрунтованим, так і максимально наближеним до реальних сценаріїв застосування ШІ.

2.2 Формалізація структури експерименту

Для забезпечення об'єктивного та відтворюваного порівняльного аналізу великих мовних моделей необхідно чітко визначити структуру експерименту. Формалізація цієї структури дозволяє задати послідовність дій, правила підготовки вхідних даних, способи генерації відповідей та порядок їх оцінювання. Від точності та прозорості цих кроків залежить достовірність результатів, можливість їхньої інтерпретації та подальше порівняння між моделями.

Оскільки в дослідженні розглядаються три мовні моделі – GPT, LLaMA та BLOOM – важливо забезпечити однакові умови тестування для кожної з них. Це передбачає однаковий формат запитів, уніфіковану структуру вхідного контексту, фіксовані параметри генерації тексту та єдиний набір запитань із датасету Natural Questions. Такий підхід дозволяє уникнути викривлення результатів через нерівномірні умови виконання та зосередитися на якості самої генерації.

Окрема увага приділяється способу подачі запитів до моделей. Для кожного прикладу буде сформовано вхід, який складається із текстового контексту (якщо він надається), запитання та службової інструкції, що спрямовує модель на формування відповіді у бажаному форматі. Стандартизація промптів є важливою складовою, оскільки навіть незначні зміни у формулюваннях можуть впливати на якість відповіді.

Датасет Natural Questions є комплексною структурованою колекцією даних, створеною для оцінювання систем питання-відповіді в умовах, наближених до реального використання. Його основною особливістю є те, що запитання взяті з реальних пошукових запитів Google, а відповіді базуються на вмісті статей із Wikipedia. Кожен приклад у датасеті має

чітку структуру та включає низку ключових полів, що дозволяють здійснювати гнучку обробку як у генеративному, так і в екстрактивному форматі.

До основних полів прикладу належать: унікальний ідентифікатор запитання (`example_id`), текст самого запитання (`question_text`), контекст у вигляді повного тексту Wikipedia-статті (`document_text`), а також розмітка позицій відповіді у вигляді індексів токенів. У випадках, де відповідь присутня, додатково вказуються координати початку і кінця довгої відповіді (`long_answer`), а також коротка відповідь у вигляді списку рядків (`short_answers`). Якщо відповідь відсутня, відповідне поле буде позначене як «null».

Довга відповідь вказує на фрагмент тексту, який, на думку розмітника, найбільш повно розкриває зміст запитання. Це зазвичай абзац або кілька абзаців із енциклопедичного тексту. Коротка відповідь – це стисла форма фактичної відповіді, яка може бути представлена одним словом, датою, числом або іменем. Її точність важлива для тестування здатності моделі до чіткої фактологічної генерації.

Контекст для запитання подається у вигляді розміченого HTML-подібного тексту, що містить внутрішні структури (наприклад, `Table`, `Paragraph`, `List`). Для цілей експерименту цей текст попередньо обробляється – HTML-розмітка видаляється або замінюється на простий текст, щоб забезпечити сумісність із мовними моделями, які не розпізнають структуровані теги.

Цей формат дозволяє гнучко працювати з даними: обирати лише необхідні поля для генерації, формувати уніфіковані запити для моделей, а також точно зіставляти отриману відповідь із еталонною. Така структура забезпечує основу для подальшої побудови експерименту, включаючи генерацію промптів, порівняння відповідей і проведення автоматичної оцінки.

Для побудови експерименту в межах даного дослідження було вирішено використовувати обидва типи відповідей, що доступні в датасеті Natural Questions – як довгу, так і коротку. Такий підхід дозволяє оцінити здатність моделей не лише формулювати точні фактологічні твердження, але й створювати повні, зв'язні та інформативні відповіді, які відповідають контексту і стилістичним вимогам завдання. Використання двох варіантів референсів дозволяє провести багаторівневий аналіз якості генерації.

Коротка відповідь обрана як базовий еталон для оцінки здатності моделі виділяти ключову інформацію. Вона добре підходить для точних метрик на рівні слів і фраз, таких як BLEU, ROUGE або METEOR, оскільки містить лаконічний фрагмент тексту без зайвих вставок. Такий тип відповіді є типовим для систем швидкого пошуку фактів і дозволяє об'єктивно порівнювати результати між моделями у вузькому форматі.

Водночас довга відповідь служить як референс для оцінювання здатності моделі до розширеної генерації, яка враховує контекст, логіку викладу та загальну завершеність висловлення. Вона дозволяє протестувати можливості генеративних моделей у більш складних сценаріях, де важлива не тільки правильність, але й повнота та стилістична природність відповіді. Саме на таких прикладах можна найбільш повно реалізувати застосування семантично орієнтованих метрик, зокрема BERTScore або MoverScore.

Обидва типи відповіді використовуватимуться паралельно: згенеровані моделі відповіді буде порівняно окремо з короткою та довгою референсною відповіддю. Це дозволить визначити, яку з форм представлення інформації модель відтворює краще, та чи існує залежність між якістю відповіді й типом еталону. Такий підхід також дозволяє виявити ситуації, коли відповідь моделі формально не збігається з жодною з референсних, але змістовно наближена до обох.

Таким чином, вибір одночасного використання короткої та довгої відповіді забезпечує багатовимірність аналізу й дозволяє охопити різні

аспекти генеративної поведінки моделей – від точності до змістовності, від факту до пояснення. Це дає змогу побудувати більш об'єктивну картину якості мовної генерації у завданнях питання-відповіді.

Вхід до кожної мовної моделі формуватиметься у вигляді текстового запиту, що поєднує контекст та запитання, подані в уніфікованій структурі. Такий підхід забезпечує однаковість умов тестування для всіх моделей та дозволяє мінімізувати вплив формулювання промπτу на результат генерації. Формат запиту буде максимально простим, зрозумілим і таким, що відповідає стилю інструкцій, які зазвичай використовуються для генеративних моделей.

Кожен вхідний запит міститиме дві основні частини: контекст та запитання. Контекст буде формуватися на основі довгої відповіді з датасету Natural Questions, тобто як фрагмент Wikipedia-статті, який, згідно з розміткою, містить достатньо інформації для формування правильної відповіді. Якщо модель отримуватиме повний документ (або більший обсяг), це може ускладнити завдання через надлишок нерелевантної інформації, тому доцільним є подання лише релевантного уривку, який охоплює довгу відповідь і її контекст.

Формат запиту для всіх моделей наведено на лістингу 2.1.

Лістинг 2.1 – Формат запиту до моделей

```
Context:  
{текст із Wikipedia (уривок із long answer)}  
Question:  
{текст запитання}  
Answer:
```

Усі змінні у цьому шаблоні будуть замінені на конкретні значення з відповідного прикладу в датасеті. Завдяки такій структурі модель одразу отримує вхід, який чітко розмежовує інформаційні блоки, стимулюючи її зосередитися на поставленому запитанні в межах заданого контексту.

Використання однакового шаблону промпту для всіх моделей дозволяє уникнути варіативності, що могла б виникнути через різні формати інструкцій, та забезпечити чесність порівняння результатів. Такий формат легко реалізується у коді та може бути масштабовано на велику кількість прикладів без втрати послідовності.

Для забезпечення коректності та відтворюваності експерименту всі моделі запускатимуться в однакових умовах, із фіксованими параметрами генерації та уніфікованим форматом вхідних даних. Це дозволяє мінімізувати вплив зовнішніх факторів на якість відповіді та зосередитися на реальних відмінностях у поведінці моделей.

Модель GPT (у варіанті GPT-3.5 або GPT-4) буде використовуватись через API OpenAI. Вхід передаватиметься у вигляді промпту у форматі, описаному раніше. Для генерації відповіді встановлюється параметр `temperature = 0.7`, що дозволяє моделі зберігати варіативність, але без надмірної випадковості. Також буде задано `max_tokens = 256`, що є достатнім для формування як короткої, так і довгої відповіді, не обмежуючи модель штучно. Інші параметри, такі як `top_p` і `frequency_penalty`, залишаються за замовчуванням. Всі запити виконуються через централізований інтерфейс, що дозволяє фіксувати час відповіді та зберігати отримані результати автоматично.

Моделі LLaMA та BLOOM будуть запускатися локально або через хмарне середовище з використанням бібліотеки HuggingFace Transformers. Обидві моделі працюють у генеративному режимі (`generate()`), з аналогічними параметрами: `temperature = 0.7`, `max_length = 256`, `do_sample = True`. Встановлення однакових параметрів генерації для всіх трьох моделей дозволяє здійснювати чесне порівняння результатів без упередженості, зумовленої технічними обмеженнями однієї з моделей.

Важливо зазначити, що перед кожним запуском усі моделі отримують один і той самий текст запиту, що виключає варіативність на рівні інструкції. Моделі LLaMA і BLOOM будуть попередньо завантажені

у стандартній конфігурації без додаткового донавчання – це дозволяє оцінити їхню базову продуктивність. Усі запуски будуть здійснюватися у стабільному середовищі, що гарантує однакову швидкість та розмір вхідних токенів.

Завдяки уніфікованим параметрам і контролю за середовищем виконання буде забезпечено технічну однорідність експерименту, що дозволить сфокусуватися на змістовних відмінностях у відповіді кожної моделі. Це створює надійну основу для подальшого порівняння результатів за обраними метриками.

Для подальшого аналізу, оцінювання та порівняння результатів роботи моделей, усі згенеровані відповіді зберігатимуться у структурованому форматі CSV-файлу. Такий підхід забезпечує простоту перегляду, сортування та імпортування даних у системи аналізу, а також дозволяє легко інтегрувати результати з модулями для обчислення метрик якості. CSV-формат є універсальним і підтримується практично всіма інструментами для обробки табличних даних, зокрема Python (pandas), Excel, Google Sheets тощо.

У кожному рядку таблиці буде представлено один приклад, тобто одне запитання з датасету Natural Questions та відповідь, згенерована конкретною моделлю. Структура файлу включатиме кілька обов'язкових колонок: `example_id` (ідентифікатор прикладу), `question` (текст запитання), `context` (використаний фрагмент Wikipedia-статті), `model_name` (назва моделі, яка генерувала відповідь), `generated_answer` (відповідь моделі), `short_reference` (еталонна коротка відповідь із датасету), `long_reference` (еталонна довга відповідь), а також `timestamp` (час генерації).

Додатково можуть бути включені колонки для зберігання результатів обчислених метрик – таких як `bleu_score`, `rouge_score`, `bertscore` тощо, – після завершення відповідного етапу оцінювання. Це дозволить об'єднати процес генерації й аналізу в одному місці, без необхідності створення окремих файлів для кожної моделі або метрики.

Зберігання відповідей у такому вигляді дає змогу також виконувати фільтрацію прикладів за моделлю, якістю відповіді, типом помилки чи іншими параметрами. Наприклад, можна легко виділити випадки, коли модель надала відповідь, але вона не збігається з коротким еталоном, або коли модель взагалі не згенерувала релевантного фрагменту.

Завдяки структурованому формату збереження, результати експерименту будуть готові до подальшої обробки, візуалізації та формування висновків щодо продуктивності моделей. Такий підхід забезпечує прозорість і відтворюваність експерименту, що є критично важливим у наукових дослідженнях.

Після завершення етапу генерації відповідей усі результати повинні бути підготовлені до автоматичного оцінювання з використанням заздалегідь обраних метрик якості. Для цього необхідно привести як відповіді моделей, так і референсні (еталонні) відповіді до уніфікованого формату, який буде прийнятним для подальшого обчислення показників BLEU, ROUGE, METEOR, BERTScore та MoverScore. Правильна підготовка даних є ключовим фактором для забезпечення точності та коректності підрахунку значень метрик.

Першим кроком є нормалізація тексту. Вона включає приведення всіх відповідей до нижнього регістру, видалення знаків пунктуації, додаткових пробілів та, за потреби, – стоп-слів. Це дозволяє уникнути ситуацій, коли різниця у формативанні призводить до штучно занижених результатів. Залежно від вимог конкретної метрики, текст може бути токенований або переданий у вигляді окремих слів чи фраз.

Усі відповіді моделей будуть порівнюватися як із короткими, так і з довгими референсами. Для кожної пари «згенерована відповідь – еталонна відповідь» буде обчислено набір метрик. У випадках, де відповідь моделі порожня або неінформативна (наприклад, повтор запитання або фраза типу «I don't know»), такі приклади буде позначено окремо для можливого виключення з підсумкової статистики або аналізу як помилки генерації.

Для моделей, що працюють у генеративному режимі, відповіді зазвичай не збігаються дослівно з еталонними. Тому особливу увагу буде приділено семантичним метрикам – BERTScore та MoverScore, які порівнюють сенс і контекст, а не лише точну форму. Ці метрики потребують попереднього завантаження трансформерної моделі для обчислення векторних подібностей, що також буде реалізовано в процесі підготовки.

Кінцева структура даних перед оцінюванням міститиме згенеровану відповідь, короткий еталон, довгий еталон, а також технічні поля, що визначають, чи була відповідь згенерована успішно, чи була вона порожньою, та яка модель її створила. Завдяки такій підготовці можна буде автоматизовано обчислити всі метрики в одному проході, зберегти результати в зведену таблицю та використовувати їх для подальшого аналізу ефективності моделей.

Для забезпечення репрезентативності експерименту, а також з урахуванням технічних та фінансових обмежень, було прийнято рішення сформувати випадкову тестову вибірку із загального обсягу датасету Natural Questions. Основним критерієм для визначення кількості прикладів стала можливість безкоштовного використання обраних мовних моделей або обмеженого API-доступу без потреби у додаткових платних ресурсах.

Зокрема, модель GPT (у версії GPT-3.5) використовується через OpenAI API, який має певні обмеження в межах безкоштовного використання, як за кількістю токенів, так і за кількістю запитів. Водночас, запуск моделей LLaMA та BLOOM здійснюється локально з використанням HuggingFace Transformers, що не передбачає грошових витрат, але обмежується доступною оперативною пам'яттю та обчислювальними ресурсами. Зважаючи на ці фактори, оптимальним було обрано вибірку з 100 випадкових прикладів, які включають як короткі, так і довгі еталонні відповіді.

Кожен приклад містить повний текст запитання, уривок із Wikipedia (який включає довгу відповідь), а також відповідні короткі й довгі еталонні відповіді. Формування вибірки здійснюється випадковим чином, без попереднього фільтрування за тематикою чи складністю, що дозволяє забезпечити збалансованість набору як з точки зору типів запитань, так і структури контексту. Це створює умови для об'єктивного тестування здатності моделей справлятися з широким спектром запитів.

Обраний обсяг вибірки є достатнім для якісного порівняння моделей, аналізу відповідей та виявлення відмінностей у продуктивності. Він також дозволяє оперативно проводити тестування та обчислення метрик без значного навантаження на ресурси, що важливо на стадії прототипування та первинної перевірки гіпотез. У разі потреби, ця вибірка може бути згодом розширена або адаптована під конкретні цілі додаткових експериментів.

У процесі формалізації структури експерименту було визначено ключові етапи, що забезпечують відтворюваність, об'єктивність та наукову коректність дослідження. Зокрема, було обґрунтовано використання датасету Natural Questions, який містить як короткі, так і довгі еталонні відповіді, що дає змогу оцінювати здатність моделей як до точного відтворення фактів, так і до формування повних, змістовних відповідей. Така побудова дозволяє здійснити багаторівневий аналіз ефективності мовних моделей у завданнях типу «питання-відповідь».

Особлива увага була приділена вибору полів із датасету, які будуть використані у формуванні вхідного запиту. Ретельне структурування вхідних даних, у тому числі нормалізація контексту та стандартизація промптів, забезпечує однакові умови тестування для всіх моделей. Формат запиту був спроектований таким чином, щоб максимально чітко відокремлювати контекст від запитання, створюючи умови для релевантного генерування відповіді.

Умови запуску моделей також були чітко окреслені. GPT використовується через API OpenAI із фіксованими параметрами генерації, а моделі LLaMA та BLOOM – у середовищі HuggingFace Transformers. Для всіх моделей встановлено уніфіковані параметри температури, максимальної довжини відповіді та режиму генерації, що дозволяє уникнути упереджених переваг однієї з моделей.

Було визначено формат збереження результатів: усі відповіді моделей, разом з відповідними запитаннями та еталонними відповідями, зберігатимуться у CSV-файлі зі структурованими полями. Це дозволяє організувати зручне подальше обчислення метрик якості, а також швидке сортування та аналіз прикладів. Крім того, закладено підготовку до автоматичного оцінювання, що включає нормалізацію тексту та підтримку метрик як на лексичному, так і на семантичному рівнях.

Обсяг тестової вибірки був підібраний з урахуванням обмежень безкоштовного доступу до моделей. Випадкова вибірка з 100 прикладів дозволяє отримати якісну порівняльну картину без надмірних витрат ресурсів. Такий підхід забезпечує як технічну ефективність, так і наукову обґрунтованість результатів експерименту.

Таким чином, на етапі формалізації структури експерименту було закладено фундамент для коректного, стабільного та прозорого порівняльного аналізу мовних моделей у задачах питання-відповіді. Це дозволить перейти до безпосереднього проведення експерименту та подальшого аналізу результатів.

2.3 Побудова схеми оцінювання результатів

Побудова схеми оцінювання результатів є важливим етапом дослідження, оскільки саме вона дозволяє об'єктивно та формалізовано визначити, наскільки якісно моделі справляються із завданням генерації відповіді на запитання. Основна мета цього етапу – розробити систему, яка

б дозволила зіставити згенеровану відповідь із еталонною, отримати числові оцінки за заздалегідь обраними метриками та проаналізувати продуктивність кожної моделі у вимірюваних аспектах.

Схема оцінювання передбачає використання кількох незалежних метрик, що покривають різні аспекти якості тексту. Частина з них, як-от BLEU, ROUGE, METEOR, оцінюють збіг слів, фраз або фрагментів між згенерованою відповіддю та еталоном. Ці метрики добре підходять для коротких відповідей або тоді, коли точність формулювань критично важлива. Однак у випадках, коли відповіді моделей не збігаються дослівно, але передають правильний зміст, доцільно використовувати семантично орієнтовані метрики, зокрема BERTScore або MoverScore, які оцінюють подібність змісту на рівні значень, а не лише форми.

Кожна відповідь, згенерована моделлю, буде порівнюватися як із коротким, так і з довгим еталоном. Це дозволить визначити не лише точність (наскільки модель правильно виділила факт), а й змістовність (наскільки відповідь глибоко й коректно розкриває тему запитання). Для кожної пари «відповідь – референс» обчислюватиметься набір значень за всіма метриками. Результати фіксуватимуться у таблиці, що вже підготовлена на попередньому етапі.

Оцінювання проводитиметься автоматично за допомогою спеціалізованих бібліотек, зокрема evaluate (HuggingFace), nltk.translate, rouge_score, bert_score та інших. Для кожного прикладу буде сформовано підсумковий блок оцінок, а також середні значення по кожній моделі, що дозволить здійснити глобальне порівняння. Крім того, будуть можливі візуалізації результатів, такі як гістограми, боксплоти або таблиці з підсвічуванням кращих результатів.

Особливу увагу буде приділено аномальним випадкам, коли всі метрики дають низькі оцінки, але модель фактично відповіла правильно – або навпаки, коли є високі оцінки при фактично помилковій відповіді. Такі

випадки будуть зафіксовані для подальшого ручного аналізу, що дозволить зробити висновки про межі застосування автоматичних метрик.

Таким чином, побудована схема оцінювання поєднує формалізовану автоматичну обробку результатів із можливістю поглибленого аналізу помилок, що робить її гнучкою та надійною основою для подальшого етапу – інтерпретації результатів.

BLEU (Bilingual Evaluation Understudy) – це одна з найперших і найпоширеніших автоматичних метрик для оцінки якості текстів, згенерованих машинними моделями. Спочатку BLEU була запропонована у 2002 році для оцінки якості машинного перекладу, але з часом її адаптували до різноманітних задач генерації тексту, зокрема питання-відповіді, резюмування, чат-ботів, тощо.

BLEU працює на основі порівняння послідовностей слів (n-грам) між згенерованим текстом (гіпотезою) та еталонним текстом (референсом). Вона оцінює, який відсоток фраз із відповіді моделі збігається з тими, що містяться в правильній відповіді.

Ось ключові принципи BLEU:

– N-грамна точність (n-gram precision). BLEU перевіряє, наскільки часто фрагменти (1-грам, 2-грам, 3-грам тощо) з відповіді моделі зустрічаються в еталонній відповіді. Наприклад, якщо модель згенерувала фразу «the cat is on the mat», а референс містить ті ж слова, BLEU врахує збіги одно-, дво-, три- та чотирислівних комбінацій;

– кліпінг (clipping). Щоб модель не «накручувала» бал через повтори, BLEU враховує кожну n-граму не більше ніж стільки разів, скільки вона зустрічається в еталоні. Наприклад, якщо модель 5 разів повторила «the», а в еталоні воно є лише двічі – врахуються тільки 2 входження;

– покарання за довжину (brevity penalty). Модель може генерувати дуже короткі, але точні відповіді (наприклад, лише 1–2 слова), щоб отримати високий відсоток збігів. Щоб запобігти цьому, BLEU вводить

штраф за надто короткі відповіді: якщо гіпотеза коротша за референс, її оцінка зменшується;

– комбінування n-грам. BLEU зазвичай обчислює середню геометричну точність для n-грам від 1 до 4 (BLEU-4). Це дозволяє врахувати як окремі слова, так і більш складні фрази. Загальний бал – число від 0 до 1 (або від 0 до 100 у відсотках), де 1 означає повний збіг.

Приклад:

- гіпотеза: «the cat sat on the mat»;
- референс: «the cat is sitting on the mat».

BLEU може виявити, що є високий збіг у 1-грам і 2-грам, але 3-грам і 4-грам не збігаються через відмінності в граматиці, тому підсумковий бал буде помірним.

Сильні сторони BLEU:

- швидка й проста реалізація;
- стандартизована для багатьох задач;
- добре працює в задачах з однозначним правильним результатом.

Недоліки BLEU:

- не враховує семантику (два різних формулювання однієї ідеї можуть отримати низький бал);
- погано підходить для відкритих генеративних задач, де допустимих відповідей багато;
- надмірно чутлива до синтаксичних варіацій.

У рамках цього дослідження BLEU буде використана для оцінювання точності коротких відповідей, де збіг фраз є показником коректності. У поєднанні з семантичними метриками BLEU дає змогу краще зрозуміти, чи модель «вгадала» правильну суть або просто механічно відтворила шаблон.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) – це метрика, яка, на відміну від BLEU, більше орієнтована на повноту (recall), а не на точність. Вона широко використовується для оцінки резюмування,

але також добре підходить для аналізу довших відповідей у завданнях питання-відповіді.

Основна ідея ROUGE – порахувати, які фрагменти еталонної відповіді вдалося покрити у згенерованій відповіді. ROUGE оцінює, скільки n-грам, слів або навіть цілих речень з референсу зустрічаються в відповіді моделі.

Найпоширеніші варіанти:

- ROUGE-1: збіг окремих слів;
- ROUGE-2: збіг пар слів (біграм);
- ROUGE-L: найдовша спільна підпоследовність (longest common subsequence) – враховує порядок слів.

Сильна сторона ROUGE – добре підходить для змістовного узагальнення або довгих відповідей, де важливо покрити якомога більше важливої інформації з еталону. На відміну від BLEU, який «карає» за зайве, ROUGE «нагороджує» за покриття.

Високий ROUGE означає, що модель охопила основні ідеї з еталонної відповіді, навіть якщо не дослівно. Це робить метрику корисною в нашому дослідженні для аналізу довгих відповідей моделей.

METEOR (Metric for Evaluation of Translation with Explicit ORdering) – це метрика, створена як поліпшення до BLEU. Вона враховує не лише точність і повноту збігів слів, але й синонімію, варіації словоформ та порядок слів, що робить її більш гнучкою для оцінки генеративних відповідей.

На відміну від BLEU, METEOR не обмежується лише точними збігами. Вона використовує лематизацію (наприклад, «run» і «running» – одна лема), а також WordNet для виявлення синонімів. Це дозволяє метриці враховувати семантичну схожість навіть при відсутності точного збігу.

METEOR поєднує precision (наскільки згенерована відповідь відповідає референсу) і recall (наскільки референс покривається

відповіддю), надаючи більшу вагу recall. Також враховується покарання за порушення порядку слів, тобто якщо слова з референсу є, але хаотично розміщені, результат знижується.

METEOR особливо цінна в задачах, де можливо багато варіантів правильної відповіді з однаковим змістом, що властиво і завданням питання-відповіді. Вона більш чутлива до смислової близькості, ніж BLEU або ROUGE, але все ще менш глибока за BERTScore.

У межах цього дослідження METEOR стане хорошим балансом між лексичними та семантичними метриками, особливо корисним для середньої довжини відповідей.

BERTScore – це метрика, яка оцінює семантичну подібність між згенерованим текстом і еталонною відповіддю, використовуючи векторні подання слів із трансформерних моделей, таких як BERT. На відміну від BLEU, ROUGE чи METEOR, вона не фокусується на точному збігу слів або фраз, а порівнює значення слів у контексті.

Кожне слово в реченні подається як вектор у багатовимірному просторі, що враховує його значення, контекст і взаємозв'язки з іншими словами. Потім для кожного слова в відповіді моделі знаходиться найбільш схожий вектор у еталоні, і обчислюється середнє значення косинусної подібності.

BERTScore видає три основні показники:

- Precision – наскільки зміст відповіді моделі охоплює зміст еталону;
- Recall – наскільки зміст еталону відтворено у відповіді;
- F1-score – гармонічне середнє між ними.

Основна перевага BERTScore – здатність виявляти правильні, але переформульовані відповіді, тобто ті, що не збігаються дослівно, але мають однакове значення. Це особливо цінно у завданнях типу «питання-відповідь», де допустимі кілька правильних варіантів.

Таким чином, BERTScore є ключовою метрикою для оцінки глибинного смислового збігу, і в межах нашого дослідження вона буде однією з найважливіших для аналізу як коротких, так і довгих відповідей.

MoverScore – це сучасна метрика для оцінювання якості автоматично згенерованого тексту, яка поєднує ідеї з BERTScore та методу Earth Mover's Distance. Її головна мета – виміряти семантичну подібність між згенерованою відповіддю та еталонною, навіть якщо вони суттєво відрізняються за формою.

На відміну від BLEU або ROUGE, які просто рахують збіги слів чи фраз, MoverScore працює з векторними поданнями слів (як і BERTScore), але оцінює, наскільки «далеко» слова з відповіді моделі мають «переміститися», щоб перетворитися на слова еталонної відповіді. Ця «відстань» у векторному просторі і є основою для оцінки.

Суть у тому, що MoverScore:

- розглядає текст як набір смислових одиниць;
- розраховує, наскільки дорого одне «значення» перетворити в інше;
- чим менше зусиль потрібно на «переміщення», тим більша схожість між відповідями.

Метрика враховує частотність слів, контекст, розмір обох текстів і видає підсумковий рейтинг семантичної близькості (зазвичай від 0 до 1). Вона особливо корисна тоді, коли відповіді довгі, вільно сформульовані або використовують синонімію.

У задачі питання-відповіді MoverScore дозволяє оцінити, чи донесла модель ту ж саму ідею, навіть якщо використала інші слова або порядок. У нашому дослідженні вона стане ще однією важливою семантичною метрикою, що доповнює BERTScore.

У результаті побудови схеми оцінювання було визначено комплекс метрик, що дозволяють всебічно проаналізувати якість відповідей моделей у задачі питання-відповіді. Було обрано як лексичні (BLEU, ROUGE, METEOR), так і семантичні (BERTScore, MoverScore) метрики, які

охоплюють точність, повноту, порядок та змістовну подібність текстів. Такий підхід забезпечує об'єктивність і глибину аналізу, дозволяючи не лише зафіксувати кількісні показники, а й виявити слабкі місця оцінювання. Сформована схема є надійною основою для подальшого експериментального дослідження та порівняння мовних моделей.

2.4 Загальна архітектура експериментальної системи

У межах дослідження важливо забезпечити чітку організацію процесу обробки даних, запуску моделей, збору результатів та їх автоматичного аналізу. Це вимагає побудови узгодженої архітектури експериментальної системи, яка б охоплювала всі етапи – від підготовки вхідних даних до фінального збереження метрик якості. Архітектура має бути достатньо гнучкою для підтримки різних моделей, форматів відповідей і варіантів подання результатів, а також мати модульну структуру для подальшого масштабування або модифікацій.

Експериментальна система реалізується як послідовність взаємопов'язаних компонентів, кожен з яких виконує специфічну функцію в межах загального експериментального процесу. Такий підхід забезпечує відокремлення логіки генерації, оцінювання та агрегації результатів, що сприяє повторюваності дослідження та підвищенню прозорості всіх операцій. Усі складові архітектури взаємодіють через визначені інтерфейси, що дозволяє легко змінювати конфігурацію системи або додавати нові компоненти без суттєвої перебудови всього процесу.

Для полегшення розуміння загальної логіки функціонування експериментальної системи було створено блок-схему, яка наочно демонструє послідовність основних етапів дослідження. Вона відображає процес від початкового завантаження даних до генерації відповідей, їх збереження, обчислення метрик та формування підсумкових результатів.

Завдяки графічному поданню можна легко простежити взаємозв'язки між компонентами системи, зрозуміти, які саме дії виконуються на кожному етапі, та як передається інформація між модулями. Це сприяє кращій структурованості експерименту та підвищує прозорість усіх процедур дослідження. Схема наведена на рисунку 2.1.

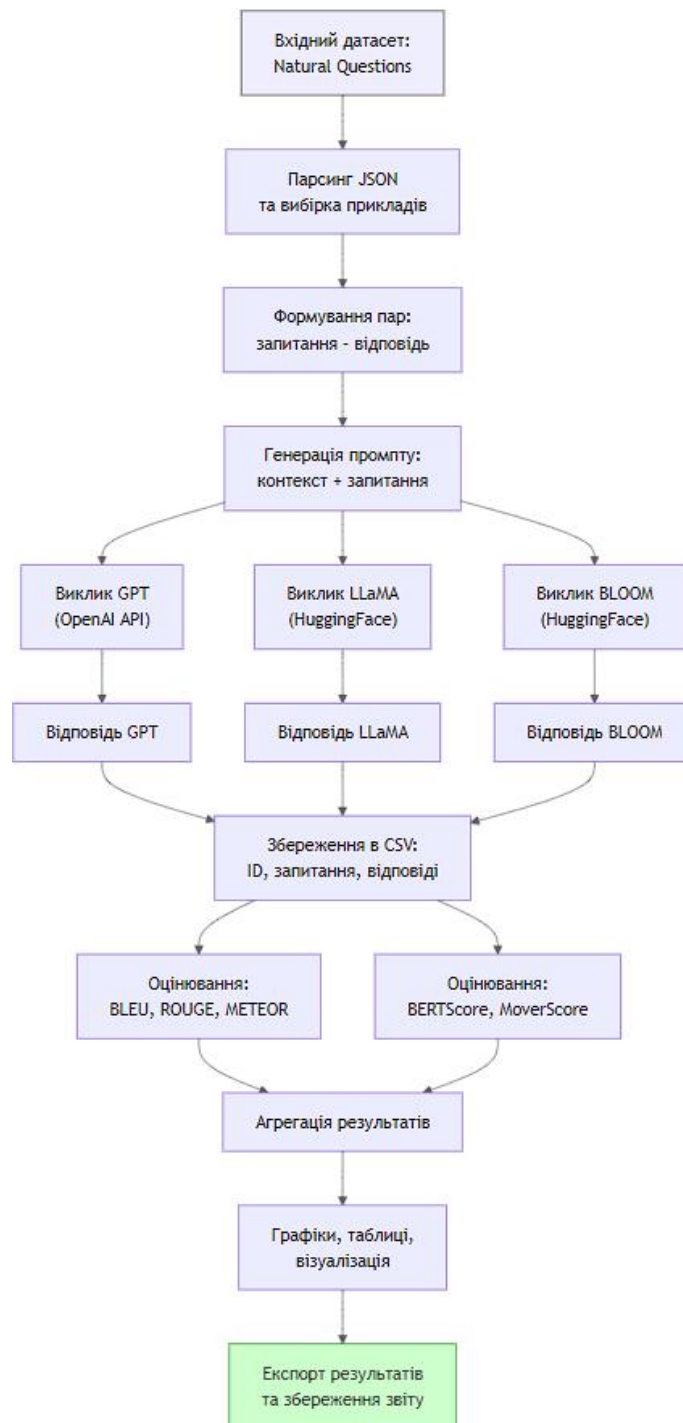


Рисунок 2.1 – Блок-схема архітектури експерименту

Представлена блок-схема відображає загальну архітектуру експериментальної системи, що реалізована для дослідження великих мовних моделей у задачі питання-відповіді. Схема демонструє основні етапи роботи системи, починаючи з підготовки вхідних даних і закінчуючи формуванням звіту з результатами оцінювання.

На початку процесу виконується завантаження вхідного датасету Natural Questions. Далі система переходить до парсингу JSON-файлу з даними та вибірки потрібної кількості прикладів. На основі кожного прикладу формується пара «запитання – відповідь», яка буде використана для генерації вхідного пром프트. Промпт створюється у стандартному вигляді з контексту (витягу з Wikipedia) і запитання.

Далі цей промпт одночасно подається трьома моделями: GPT (через OpenAI API), LLaMA і BLOOM (через HuggingFace Transformers). Кожна з моделей генерує відповідь на основі заданого контексту, і результати передаються до модуля збереження. Усі відповіді, разом із запитаннями, ідентифікаторами прикладів та назвами моделей, записуються у файл формату CSV.

Після збору відповідей розпочинається етап автоматичного оцінювання. Для цього використовується дві групи метрик: лексичні (BLEU, ROUGE, METEOR) та семантичні (BERTScore, MoverScore). Кожна відповідь моделі порівнюється як із короткою, так і з довгою еталонною відповіддю, після чого результати обробляються та агрегуються – обчислюються середні значення, виявляються крайні випадки, формується зведена таблиця.

Останній етап передбачає створення графіків, таблиць і візуалізацій для представлення отриманих результатів у зручному вигляді. Це дозволяє інтерпретувати поведінку моделей, виявити закономірності та сформулювати висновки щодо їх ефективності. У фіналі здійснюється експорт зведених результатів і збереження звіту, який стане основою для подальшого аналізу та включення у дипломну роботу.

3 РЕАЛІЗАЦІЯ ЕКСПЕРИМЕНТАЛЬНОГО ДОСЛІДЖЕННЯ ТА АНАЛІЗ ЯКОСТІ ВІДПОВІДЕЙ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

3.1 Опис реалізації генерації відповідей

3.1.1 Запуск моделі BLOOM

Модель BLOOM (BigScience Large Open-science Open-access Multilingual Language Model) є однією з наймасштабніших відкритих ініціатив у галузі великих мовних моделей. Вона була створена у рамках проєкту BigScience, який об'єднав сотні дослідників з усього світу з метою розробки прозорої, доступної та багатомовної альтернативи комерційним моделям. BLOOM – це каузальна трансформерна модель, орієнтована на генерацію тексту, яка здатна продовжувати послідовність символів, слів або речень на основі вхідного запиту.

Архітектурно BLOOM реалізована на основі стандартного підходу до autoregressive-моделей, аналогічно до GPT: вона генерує кожне наступне слово, опираючись на попередній контекст. Модель була навчена на понад 350 мільярдів токенів із понад 40 мов, що робить її однією з найуніверсальніших відкритих моделей свого часу. Завдяки цьому BLOOM добре працює не лише з англійською, але й з багатьма іншими мовами, хоча в межах цього дослідження використовується виключно для англомовних прикладів.

Для запуску BLOOM не потрібна API-підписка або спеціалізовані сервери – її можна завантажити та використовувати локально через платформу HuggingFace Transformers, що робить її зручною для наукових експериментів. HuggingFace надає попередньо навчені ваги моделей у різних розмірах – від BLOOM-560M до BLOOM-176B, з можливістю адаптації до доступних обчислювальних ресурсів. Для запуску на звичайній машині зазвичай використовують полегшені версії, наприклад

BLOOM-1B1 або BLOOM-1B7, які можна обробляти на GPU із 12–16 ГБ пам'яті.

Запуск моделі потребує встановлення бібліотек `transformers` та `torch`, а також завантаження відповідного токенизатора. Для отримання відповіді необхідно сформулювати вхідний текст у вигляді промпту, який подається моделі у функцію `generate`. Процес генерації може бути налаштований за допомогою параметрів, таких як `max_new_tokens`, `temperature`, `top_k` тощо, що впливають на довжину та варіативність згенерованого тексту.

Таким чином, BLOOM є повноцінною генеративною мовною моделлю з відкритим кодом, яка надає широкі можливості для дослідницьких експериментів без залежності від закритих рішень. У межах цієї роботи вона використовується як одна з базових моделей для порівняння якості відповідей у задачі питання-відповіді, і демонструє, наскільки ефективною може бути відкрито доступна альтернатива комерційним системам.

Для забезпечення коректного запуску моделі BLOOM у локальному середовищі було проведено підготовку програмного оточення з використанням сучасних інструментів Python-екосистеми. Основним завданням цього етапу було встановлення всіх необхідних бібліотек для завантаження моделі, токенизації вхідних даних, генерації відповідей та подальшої обробки результатів.

Насамперед, було встановлено платформу HuggingFace Transformers, яка є стандартом де-факто для роботи з великими мовними моделями у відкритому доступі. Вона забезпечує високорівневий інтерфейс для взаємодії з BLOOM, включаючи завантаження моделей з репозиторію, використання попередньо навчених токенизаторів та реалізацію функцій генерації. Для інсталяції використовувалась команда `«pip install transformers»`.

Оскільки модель BLOOM для ефективної роботи потребує підтримки GPU, було також встановлено бібліотеку PyTorch із CUDA-інтеграцією. Її

версія підбиралась відповідно до конфігурації відеокарти та версії драйвера NVIDIA. Для встановлення використовувалась офіційна інструкція з сайту PyTorch із зазначенням відповідного CUDA-білду.

Окрім того, для роботи з модельними конфігураціями та збереження результатів була встановлена бібліотека datasets від HuggingFace, яка полегшує обробку стандартних форматів датасетів. У межах обробки Natural Questions ця бібліотека могла використовуватись як допоміжний інструмент для попереднього аналізу.

Також було встановлено бібліотеки для обробки тексту, збереження CSV-файлів, логування та візуалізації результатів: pandas, numpy, tqdm, matplotlib. Вони не є критично важливими для самої генерації, але суттєво спрощують подальшу роботу з відповідями та метриками.

Таким чином, сформоване середовище включає набір перевірених інструментів, які забезпечують повноцінну роботу з моделлю BLOOM у локальному режимі, з можливістю масштабування обсягу генерації та інтеграції з іншими компонентами експериментальної системи.

На початковому етапі реалізації експериментального середовища для дослідження якості великих мовних моделей було обрано модель BLOOM у конфігурації 1,7 мільярда параметрів. Такий вибір був зумовлений тим, що дана модель займає відносно мало місця на диску та може бути запущена локально без необхідності використання спеціалізованих GPU-обчислень. Крім того, вона підтримується платформою Hugging Face Transformers, що значно спрощує її інтеграцію у програмне середовище експерименту. Передбачалося, що навіть полегшена версія моделі BLOOM зможе впоратися з базовими запитаннями із датасету Natural Questions за умови правильно сформованого промпту (рисунки 3.1).

Для перевірки цієї гіпотези було реалізовано скрипт, який виконував попередню обробку даних: виділення контексту, запитання, побудову шаблону промпту, генерацію відповіді, а також подальше збереження результатів у форматі JSON. Формат вхідних даних передбачав структуру

«Context: [текст] Question: [запитання] Answer:», після чого модель мала згенерувати релевантну відповідь.

```
prompt = (  
    f"Given the context below, answer the question.\n\n"  
    f"Context:\n{context}\n\n"  
    f"Question: {question}\n"  
    f"Answer:"  
)
```

Рисунок 3.1 – Структура промπτу

Однак після запуску перших тестових запитів результат виявився незадовільним. Модель BLOOM-1b7 не змогла адаптуватися до поданої інструкції. У ряді випадків вона просто повторювала частини контексту, не надаючи жодної відповіді на поставлене питання. Іншими словами, модель не демонструвала ознак розуміння задачі витягування інформації, що лежить в основі формату question answering.

Детальний аналіз виведених відповідей показав, що модель не тільки не виконує потрібного завдання, але й часто ігнорує структуровану форму промπτу. У відповідях фігурували або уривки з оригінального контексту, або повторення самого запитання. Наприклад, при поданні питання про найвідомішу футбольну команду Іспанії, модель могла відповісти повним запитанням або першими реченнями з контексту, жодним чином не сформувавши цільової відповіді «Real Madrid» (рисунок 3.2).

```
{  
  "question": "benefits of colonial life for single celled organisms",  
  "context": "Colony ( Biology ) - wikipedia <H1> Colony ( Biology ) </H1> Jump to : navigation , search <P> In biology , a colony is composed of two or more conspecific individuals living in c  
  "reference_short": "",  
  "reference_long": "",  
  "bloom_answer": "Colonies allow a single colony organism to benefit from the collective work and resources of the whole colony\n\nWell, if you want to ask how the colonists benefit, you might
```

Рисунок 3.2 – Незнатність моделі до обробки контексту

Окрім цього, у багатьох випадках модель взагалі не реагувала на частину промπτу з Answer:. Вона або ігнорувала її, або трактувала як частину тексту, яку слід повторити. Це ще раз підтвердило, що модель не навчена на інструкційних даних і не здатна імітувати поведінку інструкційно-орієнтованих LLM.

Також спостерігалось перенасичення відповіді контекстом. У деяких випадках у полі bloom_answer модель генерувала значну частину початкового тексту, з якого було сформовано context, наче намагаючись «відновити» вхідну частину промπτу замість того, щоб її проаналізувати. Це створювало хибне враження, що модель начебто працює, хоча фактично вона лише копіювала вже подану інформацію.

Ще однією проблемою стала відсутність узгодженості структури відповідей. Оскільки у результатах відсутній чіткий поділ між службовою частиною інструкції та сгенерованим змістом, автоматична обробка таких результатів ставала ускладненою. Невідомо було, де закінчується інструкція, а де починається власне відповідь моделі. Це негативно позначалося на оцінюванні результатів з використанням метрик BLEU, ROUGE та BERTScore.

Попри спроби нормалізувати результат – зокрема, виділяти текст після Answer: або очищати дублікат контексту – загальна якість результатів залишалась неприйнятною. Модель демонструвала низьку здатність до генерації відповіді навіть на елементарні запитання, що вимагали знаходження одного слова чи фрази в тексті.

Окремо слід згадати, що модель не демонструвала жодних ознак логічного висновування або дедуктивного мислення. Всі відповіді базувалися лише на поверхневому повторенні відомого, а не на аналізі взаємозв'язку між контекстом і запитанням. Це свідчить про слабку контекстуальну обізнаність на рівні такого масштабу параметрів.

Крім того, навіть при спробах зменшити обсяг вхідного контексту до 1000 символів з метою зниження когнітивного навантаження на модель,

покращення не спостерігалось (рисунок 3.3). Це означає, що обмеження контексту лише допомогло усунути дублювання, однак не дало змоги моделі перейти до релевантного генеративного режиму.

```
context_tokens = [t["token"] for t in tokens if not t.get("html_token", False)]
context = " ".join(context_tokens)
context = context[:1000]
```

Рисунок 3.3 – Обмеження довжини контексту

Підсумовуючи, можна констатувати, що модель BLOOM у конфігурації 1b7 не є придатною для вирішення задач типу extractive або abstractive QA, навіть за умови правильно сформованого інтерфейсу взаємодії та оптимального обмеження контексту. Застосування її у системах, що потребують осмисленої взаємодії з інструкцією, призводить до генерації псевдозмісту, який не має цінності для користувача чи аналітика.

У результаті була обґрунтована необхідність переходу до більш складної архітектури, а саме до моделі BLOOMZ-3B, яка з урахуванням інструкційного донавчання змогла забезпечити набагато вищу якість відповідей. Її використання дозволило не лише суттєво підвищити точність генерації, але й забезпечити стабільність структури відповіді, що стало критично важливим для подальшого застосування метрик якості.

Модель BLOOMZ-3B була обрана як наступний крок у дослідженні після невдалого експерименту з BLOOM-1b7. Основною причиною такого вибору стало те, що BLOOMZ – це інструкційно-доопрацьована версія оригінальної BLOOM, яка була додатково навчена на великому обсязі інструкційних даних різними мовами. У конфігурації з трьома мільярдами параметрів вона є відносно компактною, але вже демонструє достатню потужність для виконання задач середньої складності, включно з

питаннями-відповідями, особливо якщо вхідна інструкція сформульована чітко.

На відміну від своєї попередниці, модель BLOOMZ-3B продемонструвала чітке розуміння структури інструкційного промпту. Після того як у вхідному шаблоні було подано блок із контекстом, запитанням і закінченням «Answer:», модель стабільно повертала відповідь, відокремлену від попередньої частини. Це дозволило говорити про перехід від імітації контексту до справжньої генерації змісту, релевантного запиту (рисунок 3.4).

```

"question": "what is the name of spain's most famous soccer team",
Create Jira Issue
"context": "Football in Spain - wikipedia Football in Spain Football in Spain",
"reference_short": "Real Madrid",
"reference_long": "In a survey of sports habits of the Spanish population",
"bloomz_answer": "Real Madrid"

```

Рисунок 3.4 – Коректні відповіді

Однією з важливих переваг BLOOMZ-3B стала здатність до екстрагування інформації з великого контексту. Навіть за умов, коли обсяг вхідного фрагмента перевищував 1000 символів, модель не втрачала здатності виділити релевантну частину тексту і сформулювати стиснуту відповідь, не зважаючи на дійсно великий обсяг контексту, що сягав, іноді, більше 10000 символів (рисунок 3.5). Ця властивість особливо корисна для роботи з реальними документами, де запитання може стосуватися лише однієї речення з великого обсягу тексту.

У процесі експериментального використання BLOOMZ-3B не спостерігалось систематичних повторів контексту у відповіді, що було однією з найбільших проблем у BLOOM-1b7. Навпаки, модель намагалася відповідати компактно, і навіть якщо починала з повтору частини інструкції, вона швидко переходила до суті, чітко вказуючи відповідь після

Особливо варто відзначити стабільність моделі в генерації коротких відповідей. У випадках, коли очікувалась лаконічна відповідь, наприклад назва міста, особи чи об'єкта, BLOOMZ-3B не прагнула розширити зміст і не генерувала зайвого тексту. Це важливо в системах, де відповіді використовуються для побудови подальших логічних ланцюжків, аналітики чи заповнення структурованих полів у базах даних.

У процесі аналізу результатів, згенерованих за допомогою моделі BLOOMZ-3B, було виявлено характерну особливість – переважна більшість відповідей, приблизно 95%, мають надзвичайно коротку структуру (рисунки 3.6, 3.7, 3.8). У таких випадках модель обмежується відповіддю, що складається лише з двох-п'яти слів, зосереджуючись винятково на ключовій інформації, яка безпосередньо відповідає запиту. Це дозволяє стверджувати, що BLOOMZ-3B демонструє здатність формулювати стислі, інформативні та релевантні відповіді, що відповідає найкращим практикам систем питання-відповіді.

```
{
  "question": "who sings the song i don't care i love it",
  "context": "I Love It ( Icona Pop song ) - wikipedia I Lov
  "reference_short": "Icona Pop and Charli XCX",
  "reference_long": "The song 's lyrics describe breaking up
  "bloomz_answer": "Charli XCX"
```

Рисунок 3.6 – Приклад генерації короткої відповіді

```
"question": "where is zimbabwe located in the world map",
"context": "Zimbabwe - Wikipedia Zimbabwe This is the latest a
"reference_short": "in southern Africa , between the Zambezi a
"reference_long": "Zimbabwe ( / zɪmˈbɑːbwe / ) , officially t
"bloomz_answer": "Africa"
```

Рисунок 3.7 – Приклад генерації короткої відповіді

```

"question": "when did power rangers tv show come out",
"context": "Power Rangers - wikipedia Power Rangers Jump",
"reference_short": "August 28 , 1993",
"reference_long": "Power Rangers is an American entertain",
"bloomz_answer": "1983"

```

Рисунок 3.8 – Приклад генерації короткої відповіді

Такий підхід особливо важливий у контексті автоматизованого оцінювання відповідей, оскільки коротка відповідь знижує ймовірність включення лексично або семантично зайвих елементів, що можуть спотворити метрики на кшталт BLEU чи ROUGE. Завдяки цьому спрощується подальша обробка результатів, зокрема нормалізація та зіставлення з референсами. Лаконічність відповіді також покращує користувацький досвід, оскільки дозволяє безпосередньо отримувати суть без потреби в додатковому фільтруванні інформації.

Попри переваги стислих відповідей, така поведінка моделі має й потенційні недоліки. У ряді випадків коротка відповідь не передає повноти сенсу, очікуваної користувачем. Наприклад, якщо запитання має багатозначність або потребує уточнення контексту, занадто лаконічна відповідь може бути недостатньо зрозумілою або навіть вводити в оману. У прикладах із датасету Natural Questions іноді траплялися ситуації, коли модель повертала лише ім'я або дату, хоча правильна відповідь вимагала вказання зв'язку між цими елементами, що робило коротку відповідь формально правильною, але семантично обмеженою.

Крім того, така стратегія може негативно вплинути на результати оцінювання за певними метриками. Наприклад, ROUGE чи METEOR більш прихильні до відповідей, які мають значне перетинання з еталонним текстом. Якщо модель систематично скорочує відповіді до кількох слів, навіть правильних, це знижує значення метрик, що орієнтовані на кількість спільних фраз або словоформ. Як наслідок, оцінка якості може не

відображати фактичну релевантність відповіді, що важливо враховувати при аналізі результатів.

Модель також продемонструвала помітну узгодженість у стилі генерації. Навіть за наявності нетипових або розмитих інструкцій, вона намагалася зберігати логічну структуру відповіді. Це вказує на глибше засвоєння не лише мовних закономірностей, але й формату взаємодії «людина-машина» у вигляді інструкційних діалогів.

У процесі роботи модель іноді повторювала частину промпту у відповіді, однак такі випадки траплялися рідко і, як правило, не заважали автоматизованій обробці результатів. Було запропоновано рішення – виділяти зміст після останнього входження маркера «Answer:», що дозволяло очищати відповідь до стиснутого виду. Такий підхід дозволив забезпечити уніфіковану обробку всіх результатів і суттєво знизити похибки в подальшому підрахунку метрик.

У процесі генерації відповідей за допомогою моделі BLOOMZ-3B було зафіксовано окремі випадки, коли модель не надавала чіткої змістовної відповіді, а повертала узагальнені фрази на кшталт «Answer not in context» або «Not enough information». Подібна поведінка моделі свідчить про її здатність виявляти відсутність релевантної інформації у поданому контексті, що потенційно можна вважати позитивною рисою. Проте в деяких випадках ці фрази з'являлися навіть тоді, коли відповідь очевидно була присутня в тексті, що свідчить про обмеження у глибокому розумінні чи узагальненні інформації (рисунок 3.9).

```
"bloomz_answer": "not enough information"  
"bloomz_answer": "Answer not in context"
```

Рисунок 3.9 – Приклади невдалих відповідей

Поява таких шаблонних відповідей також може бути наслідком попереднього донавчання моделі на інструкційних наборах, де подібні формулювання часто використовуються для маркування нерелевантних або недостатньо визначених запитань. У випадках, коли модель не може чітко інтерпретувати зв'язок між питанням та контекстом, вона обирає обережну стратегію – вказує на брак інформації, замість того щоб генерувати випадкову або хибну відповідь.

Однак така стратегія іноді призводить до втрати потенційно правильних відповідей. Аналіз частини таких прикладів показав, що у багатьох випадках відповідь справді містилася в контексті, однак подавалася непрямо або в неочевидній формі. Модель, ймовірно, не змогла провести достатньо глибоку семантичну прив'язку між запитом та фрагментом тексту, внаслідок чого зробила хибний висновок про відсутність інформації.

Ще одним типом помилок стали випадки, коли BLOOMZ-3B повертала відповідь, яка зовсім не відповідала ні питанню, ні змісту контексту. Наприклад, на запитання, що стосувалося історичних фактів, модель іноді генерувала відповіді, що стосувалися сучасних подій або вигаданих даних. Такі відповіді могли бути граматично коректними, проте змістовно не мали жодного відношення до вихідної інформації.

Особливо критично це сприймається в тих випадках, коли результат моделі виглядає правдоподібно, але є помилковим по суті. Це створює ризик поширення хибних тверджень, що є неприйнятним у відповідальних сферах застосування, таких як медична чи юридична експертиза. Подібні помилки складно виявити автоматичними метриками, адже вони можуть мати поверхневу лексичну схожість із референсною відповіддю.

Також траплялися випадки, коли відповідь була побудована як узагальнення або надто загальна фраза, що не давала чіткої інформації. Наприклад, у відповідь на конкретне запитання модель повертала щось на зразок «Це залежить від ситуації» або «Це складне питання», що не має

жодної фактичної цінності. Подібні генерації можна трактувати як відмову від відповіді, замасковану під розлоге твердження.

Наявність таких прикладів свідчить про те, що навіть потужні моделі на кшталт BLOOMZ-3B можуть демонструвати обмеження в розумінні контексту та формулюванні відповідей, особливо за відсутності чіткої вказівки на релевантний фрагмент у тексті. Це підкреслює важливість додаткового фільтрування або перевірки результатів, особливо в тих випадках, коли точність критично важлива. У контексті дипломної роботи такі приклади були зафіксовані та враховані як один із факторів, що впливають на достовірність загальної оцінки ефективності моделі.

Варто зазначити, що час генерації відповіді для BLOOMZ-3B був прийнятним навіть у режимі CPU. Середній час генерації однієї відповіді коливався в межах кількох секунд, що дозволяло безперервно обробити сотні прикладів у рамках одного експерименту. Це стало критично важливим з огляду на обмеження безкоштовних ресурсів і відсутність можливості використовувати комерційні API.

Ще однією перевагою стало те, що відповіді BLOOMZ-3B мали вищу когерентність із поставленим запитанням. У випадках, коли BLOOM-1b7 пропускала або ігнорувала ключові слова в запиті, BLOOMZ-3B формувала релевантну відповідь, демонструючи розуміння інтенції користувача. Це свідчить про наявність механізму внутрішнього зіставлення між семантикою запиту і контексту.

У межах експерименту також перевірялося, чи впливає довжина контексту на якість відповіді. Було виявлено, що при довжині до 1024 токенів модель не втрачала здатності ідентифікувати важливу інформацію. Це дозволило зберегти повноцінну логіку QA-завдання навіть у випадках, коли релевантна відповідь знаходилась у кінці довгого тексту.

Таким чином, BLOOMZ-3B продемонструвала очікувану продуктивність, достатню для коректного виконання типових завдань питання-відповідь у форматі Natural Questions. Вона стала придатною

моделлю для подальшого порівняння з іншими LLM – зокрема GPT-4 і LLaMA-3, у межах реалізації наступних етапів роботи

3.1.2 Запуск моделі LLaMA

LLaMA (Large Language Model Meta AI) – це серія великих мовних моделей, розроблених дослідницьким підрозділом компанії Meta з метою створення відкритої, масштабованої та високоефективної архітектури для задач обробки природної мови. Уперше представлена у 2023 році, ця лінійка моделей стала важливою віхою у розвитку відкритих LLM, що здатні конкурувати з комерційними системами на кшталт GPT. Моделі LLaMA були спеціально оптимізовані для збереження балансу між розміром, точністю та обчислювальними потребами, що зробило їх зручними для дослідницьких задач, експериментів і практичного застосування.

Архітектура LLaMA базується на трансформерному механізмі з численними внутрішніми оптимізаціями, включно з використанням SwiGLU-активацій, нормалізації RMSNorm та ефективного позиційного кодування. Це дозволяє досягати кращої якості генерації при меншій кількості параметрів у порівнянні з аналогічними моделями. Модель доступна у різних масштабах – від 7 до 65 мільярдів параметрів, причому навіть молодші версії, такі як LLaMA-7B, демонструють високу якість генерації за умови правильно сформованого запиту.

Окреме значення має серія моделей LLaMA2, що була представлена як поліпшення попередньої генерації. LLaMA2 включає як базові моделі, так і версії з інструкційним тюнінгом (наприклад, LLaMA2-Chat), які спеціально навчалися відповідати на запити у форматі «інструкція – відповідь». Таке донавчання робить ці моделі особливо ефективними у задачах питання-відповіді, діалогів та генерації текстів, де важлива чітка відповідь згідно з поставленою метою.

Для запуску моделі LLaMA необхідне середовище з підтримкою PyTorch, а також доступ до моделі через платформу Hugging Face або офіційні джерела Meta. У випадку локального розгортання мінімальним технічним вимогом відповідає система з 16–24 ГБ оперативної пам'яті або GPU з достатнім обсягом відеопам'яті (як правило, від 8 ГБ для моделей 7B у 8-бітному режимі). Крім того, запуск LLaMA вимагає встановлення допоміжних бібліотек, зокрема transformers, accelerate та sentencepiece, а також виконання авторизації для завантаження моделі з Hugging Face.

У типових сценаріях взаємодія з LLaMA здійснюється через об'єкти AutoTokenizer і AutoModelForCausalLM, які забезпечують як токенизацію вхідного запиту, так і генерацію відповіді на його основі. Завдяки оптимізації, модель може ефективно працювати навіть у CPU-режимі, хоча для великих обсягів даних або великої вибірки запитів GPU-запуск залишається бажаним.

Таким чином, LLaMA – це високопродуктивна, гнучка та відносно доступна у використанні модель, яка надає всі необхідні можливості для виконання завдань типу питання-відповіді. Вона є ідеальним кандидатом для порівняльного аналізу у межах експериментальної частини даної дипломної роботи, особливо на фоні моделей, що не проходили інструкційного донавчання.

Для виконання експериментальної частини дослідження було обрано модель LLaMA 3 – останнє покоління великих мовних моделей, розроблених компанією Meta AI. На момент реалізації роботи відкритий доступ до LLaMA 3 був наданий через платформу Hugging Face, з попередньою авторизацією та погодженням умов використання. Для цілей локального запуску було обрано модель meta-llama/Meta-Llama-3-8B-Instruct, яка є інструкційно-доопрацьованою версією середнього масштабу та дозволяє ефективно працювати з задачами питання-відповіді.

Розгортання моделі відбувалося у віртуальному середовищі Python 3.10, створеному в межах локального проекту, де попередньо було

встановлено всі необхідні залежності. Основною бібліотекою для взаємодії з LLaMA 3 стала `transformers`, яка забезпечує API для токенизатора (`AutoTokenizer`) та самої моделі (`AutoModelForCausalLM`). Оскільки модель підтримує сучасні оптимізації генерації, додатково було підключено бібліотеки `accelerate` та `torch`, що дозволяють працювати як на CPU, так і з апаратним прискоренням на GPU.

Список основних пакетів, які було встановлено перед запуском моделі, включав «`pip install transformers accelerate torch sentencepiece tqdm`».

Окрему увагу було приділено питанню авторизації. Модель `Meta-Llama-3-8B-Instruct` не є повністю відкритою – для її використання необхідно пройти погодження умов використання на платформі `Hugging Face`, після чого завантаження моделі можливе лише через обліковий запис, що має відповідні права доступу. Для цього було використано інструмент `huggingface-cli login`, що дозволив здійснити інтеграцію з персональним токеном API.

Після виконання попередніх налаштувань модель автоматично завантажувалась через виклик `AutoModelForCausalLM.from_pretrained(...)`, при цьому система самостійно підтягувала усі конфігураційні файли, параметри моделі та відповідний токенизатор. У процесі роботи модель могла зберігатися у кеші `Hugging Face` для повторного використання без потреби в повторному завантаженні.

Таким чином, розгортання LLaMA 3 не вимагало складної конфігурації, але потребувало дотримання офіційного порядку авторизації та відповідної адаптації середовища. Наявність інструкційної версії LLaMA 3 дозволила уникнути ручного тюнінгу промптів і забезпечила коректне реагування на запити у форматі «контекст–запитання–відповідь», що критично важливо для цілей даного дослідження.

У процесі виконання експериментального дослідження особливу увагу було приділено запуску моделі LLaMA, зокрема її нової модифікації – `Meta LLaMA 3`. У порівнянні з моделлю `BLOOM`, яка є повністю

відкритою і не потребує жодних додаткових дозволів, використання LLaMA виявилось більш формалізованим з точки зору доступу до моделі. Перед початком інтеграції даної моделі в експеримент було виявлено, що Hugging Face не надає можливість вільного завантаження Meta LLaMA 3 без проходження процедури реєстрації та ручного затвердження запиту користувача.

Для отримання доступу до моделі Meta LLaMA 3 було потрібно попередньо авторизуватись у системі Hugging Face та перейти на сторінку відповідної моделі. Після цього користувач повинен подати заявку на приєднання до «Gated Repository», погодившись із ліцензійними умовами розробника моделі – компанії Meta AI. Без такого підтвердження будь-які спроби автоматизованого завантаження моделі через бібліотеку transformers завершуються помилкою з кодом 403, яка свідчить про відсутність відповідних прав доступу (рисунок 3.10).



```
(llama3-env) PS E:\LLMProject> python run_llama3_experiment.py
Traceback (most recent call last):
  File "E:\LLMProject\llama3-env\Lib\site-packages\huggingface_hub\utils\_http.py", line 409, in hf_raise_for_status
    response.raise_for_status()
  File "E:\LLMProject\llama3-env\Lib\site-packages\requests\models.py", line 1024, in raise_for_status
    raise HTTPError(http_error_msg, response=self)
requests.exceptions.HTTPError: 403 Client Error: Forbidden for url: https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct/resolve/main/config.json
```

Рисунок 3.10 – Помилка доступу

Після схвалення запиту користувач отримує доступ, однак цього виявляється недостатньо для безперешкодної роботи моделі. Щоб Hugging Face CLI дозволив завантаження таких приватних репозиторіїв, необхідно створити або відредагувати наявний access-token, увімкнувши в його параметрах дозвіл на читання вмісту gated репозиторіїв. Ця дія є обов'язковою, оскільки навіть за наявності формального доступу, токен без відповідного permission-прапорця буде ігноруватись, що знову призведе до помилки доступу.

На відміну від моделей типу BLOOM, що не вимагають жодного додаткового конфігурування доступу, у випадку з LLaMA реалізація доступу є більш обмеженою й чітко контрольованою. Це пов'язано не лише з ліцензійною політикою Meta, а й із необхідністю захисту комерційно орієнтованих моделей, які базуються на сучасних трансформерних архітектурах і можуть використовуватись у продуктивних середовищах.

Слід зазначити, що така структура доступу не є критично складною, однак вимагає від дослідника додаткової уважності при роботі з Hugging Face Hub, особливо при автоматизованому завантаженні моделей у скриптах. У разі недотримання інструкцій – наприклад, використання токена без доступу до gated моделей – виникають складно діагностовані помилки, що значно уповільнюють процес дослідження.

Таким чином, при запуску моделі LLaMA було необхідно не лише завантажити її, а й пройти через кілька етапів перевірки та ручної конфігурації. Водночас така процедура дозволяє компанії Meta краще відслідковувати використання своїх моделей і контролювати відповідність умовам ліцензування. Для дослідника це означає підвищення прозорості, але й певні додаткові кроки на етапі ініціалізації експерименту.

У подальших етапах дослідження, після отримання доступу, модель була успішно завантажена через API бібліотеки transformers, а її інтеграція у загальну архітектуру експериментальної системи відбувалась без технічних ускладнень. Проте саме початковий етап реєстрації і конфігурації став ключовим чинником, який відрізняв LLaMA від інших мовних моделей, що використовувались у межах цього дослідження.

Для коректної роботи з моделлю Meta LLaMA 3 через бібліотеку transformers від Hugging Face, недостатньо лише мати затверджений доступ до репозиторію моделі. Обов'язковою вимогою є також наявність авторизаційного токена (access token), який повинен мати спеціально визначені дозволи, що забезпечують доступ до так званих gated repositories.

Генерація та налаштування цього токена є критичним етапом у забезпеченні можливості взаємодії з обмеженими моделями, такими як Meta LLaMA.

Створення токена починається з входу до облікового запису на платформі Hugging Face. Далі необхідно перейти до налаштувань акаунту, у розділ «Access Tokens» (<https://huggingface.co/settings/tokens>). Тут доступний функціонал генерації нового токена. При натисканні кнопки «New token» користувачеві пропонується ввести назву токена (наприклад, llama-access) та вибрати рівень доступу.

Для роботи з LLaMA потрібно обрати тип токена «Read» (читання), який дозволяє завантаження моделей, але не надає права на модифікацію або публікацію (рисунок 3.11). Проте навіть цього недостатньо – після створення або редагування токена, необхідно вказати розширені дозволи. Для цього відкривається панель «Edit scopes» (редагування повноважень), де надаються права, які визначають доступ до специфічних типів ресурсів Hugging Face.

Ключовим параметром є «Read access to contents of all public gated repos you can access». Цей прапорець обов'язково має бути увімкненим, оскільки саме він дозволяє використовувати моделі, які вимагають додаткового погодження (наприклад, Meta LLaMA, MPT, Gemma та інші). Без цього прапорця навіть після схвалення запиту на доступ до моделі всі спроби завантаження будуть завершуватись помилкою 403 (Forbidden).

Name	Value	Last Refreshed Date	Last Used Date	Permissions
ForStudy	hf_...cnRX	5 minutes ago	4 minutes ago	FINEGRAINED

Рисунок 3.11 – Згенерований токен доступу

Також рекомендується вмикати опцію «Read access to contents of all repos under your personal namespace», що дозволяє працювати з приватними

моделями, які можуть бути збережені у власному обліковому записі. Інші параметри, такі як доступ до організацій, inference endpoints, колекцій або білінгу, у контексті цього експерименту не потрібні, і можуть залишатись вимкненими (рисунок 3.12).

Після збереження нового токена з правильними дозволами, його потрібно ввести через CLI, використовуючи команду «huggingface-cli login», після чого ввести токен доступу.

У процесі авторизації система запитає токен, який можна просто вставити в термінал. У результаті токен буде збережено локально.

Правильно налаштований токен у поєднанні з наданим доступом до моделі – це єдина умова, яка дозволяє використовувати LLaMA 3 у режимі from_pretrained(...). У разі відсутності відповідного дозволу, система не повідомляє, що справа саме у permission scope, і видає помилку, яка виглядає як типова помилка доступу, що може ускладнити діагностику для недосвідченого користувача. Саме тому правильна генерація токена з урахуванням усіх технічних вимог є критично важливою частиною підготовки до експерименту.

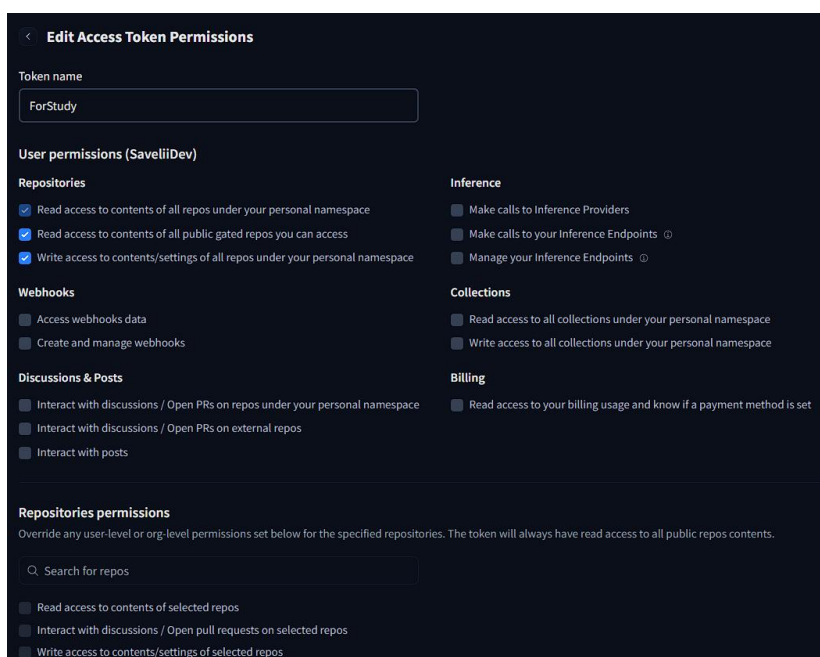


Рисунок 3.12 – Налаштування токена доступу

Під час першого запуску моделі Meta LLaMA 3 через бібліотеку transformers від Hugging Face, було ініційовано автоматичне завантаження всіх необхідних компонентів моделі з відповідного репозиторію. Цей процес є типовим для будь-якої великої мовної моделі, що зберігається в розподіленому форматі на Hugging Face Hub. У випадку з LLaMA 3 обсяг цих файлів виявився суттєвим: загальна сума завантажуваних даних перевищила 16 ГБ (рисунок 3.13).

File Name	Progress	Download Speed	Time
model-00001-of-00004.safetensors	100%	1.176/1.176	[09:50:00:00, 1.98MB/s]
model-00001-of-00004.safetensors	22%	1.106/4.986	[09:48:42:08, 1.53MB/s]
model-00001-of-00004.safetensors	75%	3.736/4.986	[24:59:07:03, 2.93MB/s]
model-00002-of-00004.safetensors	22%	1.086/5.006	[09:47:37:26, 1.74MB/s]
model-00002-of-00004.safetensors	65%	3.246/5.006	[24:59:11:51, 2.47MB/s]
model-00003-of-00004.safetensors	71%	3.476/4.926	[24:57:10:29, 2.30MB/s]

Рисунок 3.13 – Завантаження компонентів моделі

Модель зберігається у вигляді сегментованих файлів формату safetensors, які мають розширення на зразок model-00001-of-00004.safetensors, model-00002-of-00004.safetensors тощо. Цей формат є більш безпечним і ефективним у порівнянні з традиційними .bin файлами, а також дозволяє паралельне зчитування параметрів моделі під час ініціалізації. У випадку Meta LLaMA 3-8B модель розбита на 4 частини, кожна з яких має обсяг приблизно від 4 до 5 ГБ.

Під час завантаження система Hugging Face Hub автоматично зберігає ці файли у кеш-каталозі локальної системи, зазвичай у директорії ~/.cache/huggingface/transformers. Це означає, що повторне завантаження моделі в майбутньому вже не потребує доступу до інтернету, оскільки всі параметри залишаються збереженими на диску. Це важливо як для зменшення навантаження на API, так і для пришвидшення запуску подальших експериментів.

Особливістю такого першого запуску стало тривале завантаження – оскільки загальний обсяг даних склав понад 16 ГБ, при середній швидкості мережі 1.5–2 МБ/с цей процес тривав майже годину. Завантаження

відбувалося поступово по частинах, що дозволяє бібліотеці transformers одночасно обробляти потоки й показувати прогрес для кожного окремого файлу.

Таким чином, ще до початку будь-якого обчислювального експерименту з LLaMA, необхідно враховувати витрати часу та дискового простору для первинного завантаження моделі. Цей крок є обов'язковим і одноразовим, але він чітко демонструє масштаб моделі та обсяг обчислювальних ресурсів, які вона потребує для повноцінної роботи.

У процесі підготовки експериментального середовища для роботи з моделлю LLaMA було виявлено, що структура скрипту, необхідного для генерації відповідей, значною мірою повторює логіку, що раніше застосовувалась для моделі BLOOM. Це не є випадковим, адже обидві моделі використовують інтерфейс бібліотеки transformers від Hugging Face, яка забезпечує уніфікований підхід до ініціалізації, токенизації та генерації тексту. Завдяки цьому більшість базових елементів коду залишаються майже ідентичними для різних моделей, що спрощує адаптацію та повторне використання розроблених рішень.

Зокрема, основні етапи взаємодії з LLaMA 3 включають такі самі кроки: завантаження токенизатора та самої моделі через функції `from_pretrained`, формування вхідного промпту з контексту та питання, токенизація тексту, генерація відповіді за допомогою `model.generate`, а також декодування отриманих результатів у текстовий формат. Ці ж самі етапи були реалізовані для моделі BLOOM, що дозволило легко перенести існуючу логіку в новий скрипт без необхідності повної його перебудови.

Візуально структура обох скриптів майже не відрізняється. Замість назви моделі `bigscience/bloomz-3b` вказується `meta-llama/Meta-Llama-3-8B-Instruct`, а параметри генерації (наприклад, `max_new_tokens`, `temperature`, `top_p`) залишаються аналогічними. Єдиною суттєвою відмінністю є формат промпту – для LLaMA 3 рекомендується використовувати спеціальні інструкційні шаблони з маркуванням `[INST] ... [/INST]`, що є специфікою

цієї серії моделей. У скрипті BLOOM цього компонента не було, оскільки BLOOM не є інструкційною моделлю за замовчуванням.

Крім того, обробка структури датасету, зокрема витягування запитання, контексту, короткої та довгої відповіді, реалізована абсолютно однаково. Це дозволило не лише пришвидшити розробку, а й уніфікувати збереження результатів у єдиному форматі – масив JSON-об'єктів із полями `question`, `context`, `reference_short`, `reference_long` та `llama_answer` або `bloom_answer` відповідно.

Таким чином, завдяки гнучкій архітектурі бібліотеки `transformers` та попередньо реалізованому підходу до обробки даних, вдалося забезпечити високий ступінь повторного використання коду. Це значно скоротило час підготовки середовища, а також зменшило ймовірність помилок при переході від однієї моделі до іншої. У результаті вдалося досягти функціональної узгодженості експериментів на різних моделях із мінімальними змінами в програмному коді.

Під час реалізації етапу запуску моделі LLaMA було виявлено низку технічних особливостей, які суттєво вплинули на організацію експериментального процесу. Зокрема, модель виявилася найвимогливішою з-поміж усіх, що розглядалися в межах дослідження. Це стосується як обсягу системних ресурсів, необхідних для її розгортання, так і часу, що витрачається на генерацію кожної окремої відповіді. У порівнянні з іншими моделями, LLaMA вимагає значно більше обчислювальної потужності, що накладає певні обмеження на масштабованість експерименту в умовах звичайного середовища розробки.

У процесі тестування з'ясувалося, що навіть на сучасному апаратному забезпеченні, зокрема з використанням відеокарти RTX 4060 та одного з найбільш продуктивних на момент написання процесорів Ryzen 7 7800X3D, середній час генерації відповіді становив від 150 до 200 секунд (рисунок 3.14). Така тривалість обробки одного запиту істотно уповільнювала процес проведення експерименту, роблячи його менш

ефективним з точки зору витрат часу. Якщо ж говорити про менш потужні машини, то затримка могла сягати понад 30 хвилин (3.15), що робило використання моделі практично неможливим у рамках обмеженого тестового циклу.

```
Processing: 0it [00:00, ?it/s]Setting `pad_token_id` to `eos_token_id`:128009 for open-end generation.
Processing: 1it [03:20, 200.40s/it]Setting `pad_token_id` to `eos_token_id`:128009 for open-end generation.
Processing: 2it [06:37, 198.67s/it]Setting `pad_token_id` to `eos_token_id`:128009 for open-end generation.
Processing: 4it [09:48, 133.92s/it]Setting `pad_token_id` to `eos_token_id`:128009 for open-end generation.
```

Рисунок 3.14 – Швидкість обробки одного запиту на сучасній машині

```
Processing: 0it [00:00, ?it/s]Setting `pad_token_id` to `eos_token_id`:128009 for open-end generation.
Processing: 1it [32:51, 1971.59s/it]
```

Рисунок 3.15 – Швидкість обробки одного запиту на менш потужній машині

Цей факт виявився критично важливим під час прийняття рішень щодо подальшого ходу експерименту. Було зрозуміло, що без переходу на обчислювальні ресурси класу серверних або клауд-інфраструктури, обробка повного набору тестових запитів із достатнім обсягом вибірки не могла бути реалізована в розумні строки. У результаті було прийнято рішення суттєво обмежити обсяг тестових даних для цієї моделі, з метою отримання хоча б мінімального, але репрезентативного результату.

Попри вищезазначені складнощі, LLaMA демонструє високий рівень якості відповідей. Зокрема, в умовах повного доступу до контексту вона дозволяє виводити детальні, логічно послідовні відповіді, що враховують змістове навантаження попередньої інформації. Таким чином, хоча модель і не продемонструвала оперативної ефективності, її якісні характеристики залишилися на належному рівні. Проте, з точки зору практичної придатності до інтерактивного використання або автоматизованого аналізу

великих масивів даних, обмеження в часі відповіді є значним стримувальним фактором.

Таким чином, LLaMA у своїй реалізації у межах локального запуску є більш орієнтованою на дослідницьке або вузькоспеціалізоване використання, а не на широке інтегрування у високонавантажені системи реального часу. Однак саме її висока точність і здатність обробляти складні запити роблять її цінним інструментом у випадках, коли якість відповіді превалює над швидкістю отримання результату. Це ставить модель в окрему категорію, що потребує специфічних технічних умов і підходів до використання.

У ході виконання експерименту було встановлено, що модель LLaMA продемонструвала вищий рівень чутливості до контексту запитань у порівнянні з попередньо протестованою BLOOM. Це стало помітно як на рівні якісної оцінки згенерованих відповідей, так і в самому процесі взаємодії з моделлю. Особливість LLaMA, яка полягає в кращій роботі з довшими контекстами, відіграла важливу роль у загальній точності відповіді, оскільки модель була здатна глибше враховувати розширену інформацію, що передавалася разом із запитанням.

Однією з головних переваг стало те, що модель LLaMA дозволяла передавати більший обсяг контексту без втрати якості розуміння. Якщо у випадку BLOOM доводилося вручну обмежувати довжину контексту через ризик перевищення максимальної кількості токенів або спотворення значення через обриви фраз, то LLaMA стабільно обробляла контексти обсягом понад 800–1000 слів. Це дозволяло передавати майже повноцінний зміст фрагментів документів, у яких містилася відповідь, і формувати значно більш інформативний запит.

Крім того, модель LLaMA ефективніше встановлювала логічний зв'язок між фоном, запитанням та припущуваною відповіддю. Це особливо проявлялося у складних питаннях, де потрібна не лише пошукова операція, а й елементарне узагальнення або порівняння даних. У таких випадках

LLaMA демонструвала чітке опрацювання ключових тез у контексті, не ігноруючи важливих уточнень, які BLOOM у подібних ситуаціях іноді пропускала або розцінювала як другорядні.

Формат інструкційного промпту [INST] Context: ... Question: ... [/INST], що був рекомендований для LLaMA, також сприяв покращенню результатів. Завдяки чіткій семантичній структурі запиту модель мала змогу ефективно розділяти інформаційні блоки й точніше локалізувати зміст, на який слід звертати увагу під час генерації відповіді. BLOOM, яка не має інструкційного налаштування за замовчуванням, була більш схильна змішувати контекст із питанням або неадекватно визначати пріоритети в аналізі.

Підсумовуючи, варто зазначити, що можливість подавати розширені контексти стала визначальним чинником підвищення продуктивності моделі LLaMA в умовах цього дослідження. Її здатність ефективно обробляти великі обсяги вхідних даних без спотворення структури запиту й без деградації якості відповіді дозволила отримати більш точні та повні результати. Це підтверджує практичну доцільність використання LLaMA у задачах, де критично важливою є наявність детального контексту або фонові інформації.

3.1.3 Запуск моделі GPT

Модель GPT є однією з найвідоміших представників великих мовних моделей (LLM), що була створена компанією OpenAI і в короткий термін стала стандартом у галузі природньої мовної обробки. У межах проведеного дослідження було вирішено включити GPT до порівняльного аналізу, оскільки вона широко застосовується у практичних задачах питання-відповіді, має відкритий API для дослідницьких цілей, а також підтримує інструкційне форматування, яке є релевантним до завдання. Це дозволяє зіставити результати її роботи з іншими моделями, зокрема з

BLOOM та LLaMA, за однаковими умовами подачі вхідних даних і однаковими метриками оцінювання.

Особливістю GPT є те, що вона працює не як локальна модель, яку можна завантажити через Hugging Face, а як віддалений хмарний сервіс. Для її використання необхідно мати чинний API-ключ від OpenAI, який надається через особистий кабінет після реєстрації. Це, з одного боку, спрощує роботу з моделлю, оскільки не потребує завантаження великих ваг або розгортання на локальному обладнанні, а з іншого – накладає обмеження у вигляді платного доступу або квот використання.

Для інтеграції GPT у структуру експерименту було використано бібліотеку openai, яка дозволяє виконувати запити до моделей GPT-3.5 або GPT-4 у форматі Chat Completion. У порівнянні з BLOOM або LLaMA, запуск GPT не потребував ні токенизатора, ні кастомної обробки тексту на рівні токенів. Усі операції (формування промпу, його обробка та генерація відповіді) виконуються безпосередньо в межах виклику API, що значно спрощує реалізацію.

Формат взаємодії з моделлю GPT побудований на основі так званого чатового запиту, де повідомлення подаються у вигляді послідовності ролей (system, user, assistant). Для дослідження було обрано шаблон, у якому роль користувача (user) містила сформований промпт із питанням і контекстом, аналогічним до тих, що використовувалися для BLOOM та LLaMA. Це дозволило забезпечити єдність експериментальних умов і об'єктивно порівнювати результати.

Також варто зазначити, що GPT здатна ефективно обробляти значні обсяги вхідного тексту – залежно від версії моделі, межа контексту сягає 8К або навіть 32К токенів. Це дає змогу передавати повний контекст без обрізання, що особливо важливо для завдань, у яких відповідь прямо залежить від точного опрацювання всього тексту. У попередніх моделях доводилося вручну скорочувати контекст, у той час як GPT обробляла повний текст без жодної деградації якості.

На відміну від інших моделей, GPT має у своєму розпорядженні вбудовану функцію обмеження довжини відповіді та регулювання параметрів генерації, таких як `temperature`, `presence_penalty`, `frequency_penalty`, що дає змогу гнучко налаштовувати поведінку моделі без складних механізмів. Це спрощує дослідження, оскільки дозволяє швидко адаптувати модель до специфіки завдання.

Загалом, запуск GPT в рамках дослідження не вимагав таких об'ємних підготовчих дій, як у випадку з BLOOM або LLaMA. Натомість основним викликом було забезпечення правильного формування запитів і відповідне збереження результатів у такому ж форматі, як і в інших моделях. Це дало змогу забезпечити повноцінну інтеграцію GPT у загальну систему експерименту та здійснити подальше порівняння з іншими підходами.

У процесі підготовки середовища для запуску моделі GPT від OpenAI було вжито мінімальний набір дій, порівняно з іншими моделями, такими як BLOOM або LLaMA, що вимагали завантаження сотень мегабайт або навіть гігабайтів даних. GPT є хмарною моделлю, яка надається через API, тому її запуск не передбачає локального завантаження ваг чи залежностей, пов'язаних із архітектурою трансформерів. Усе обчислення виконується на серверах OpenAI, а локально потрібно лише забезпечити правильну роботу клієнтської бібліотеки.

Першим кроком стало створення окремого віртуального середовища Python, щоб ізолювати залежності GPT-модуля від інших експериментальних компонентів. Після активації середовища було встановлено бібліотеку `openai`, яка забезпечує інтерфейс для взаємодії з хмарною моделлю. Інсталяція виконувалась командою «`pip install openai`». Після цього середовище стало готовим для виконання запитів до моделі. Однією з особливостей GPT є обов'язкова вимога до наявності API-ключа, який видається OpenAI для авторизованого доступу. Для цього користувач повинен зареєструватися на платформі OpenAI, після чого отримати

персональний секретний ключ. Цей ключ або зберігається як змінна середовища `OPENAI_API_KEY`, або передається безпосередньо у скрипт як параметр конфігурації.

Крім бібліотеки `openai`, не було потреби встановлювати додаткові пакети, пов'язані з токенізацією, форматуванням вхідного тексту чи збереженням моделі. Усі ці аспекти обробляються на стороні OpenAI, що значно спрощує підготовку до експерименту. Це робить GPT зручним інструментом для дослідників, які прагнуть швидко інтегрувати модель у свою систему без необхідності мати потужну графічну карту чи локальне сховище для збереження великомасштабних параметрів.

Крім того, запуск GPT не вимагає ручного керування режимами обчислення (як-от `float16`, `int8`, `device_map=auto`), оскільки ці аспекти повністю контролюються інфраструктурою OpenAI. Це спрощує скрипт і дозволяє зосередитися виключно на формуванні правильного запиту та подальшому аналізі отриманої відповіді.

Таким чином, підготовка до запуску GPT була найменш ресурсоемною з усіх протестованих моделей, і вимагала лише встановлення одного пакета та налаштування API-доступу, що зробило цей етап технічно простим, швидким і відносно стабільним.

Модель GPT, на відміну від інших використаних у дослідженні мовних моделей, є повністю закритою та працює виключно через платний API, який надається компанією OpenAI. Для її використання необхідною умовою є отримання персонального токена доступу – секретного ключа, який ідентифікує користувача, фіксує всі запити та визначає обсяг дозволеного використання. Саме через цей механізм OpenAI здійснює облік використаних обчислювальних ресурсів та виставляє відповідні обмеження або тарифи.

Процедура отримання токена починається з реєстрації облікового запису на офіційному сайті OpenAI. Після створення акаунту користувачеві пропонується створити API-ключ, який буде надалі

використовуватись у всіх програмних запитах до моделі. Однак, попри наявність зручної інфраструктури та документації, доступ до GPT-моделі не є повністю відкритим і безкоштовним: навіть для базових запитів потрібно активувати білінг.

У ході виконання даного дослідження було виявлено, що без попереднього підтвердження платіжної інформації, включаючи внесення першого платежу, доступ до моделі GPT-3.5 або GPT-4 залишається заблокованим. Зокрема, навіть якщо в акаунті присутні стартові бонуси (так звані безкоштовні токени), система вимагає додати кредитну або дебетову карту, а також поповнити баланс на мінімальну суму – в даному випадку \$5. Це є запобіжним заходом проти масових зловживань сервісом і водночас дозволяє OpenAI активувати повноцінний доступ для програмного використання моделі. Після внесення платежу, баланс аккаунту моделі можна було подивитися в реальному часі (рисунок 3.16).

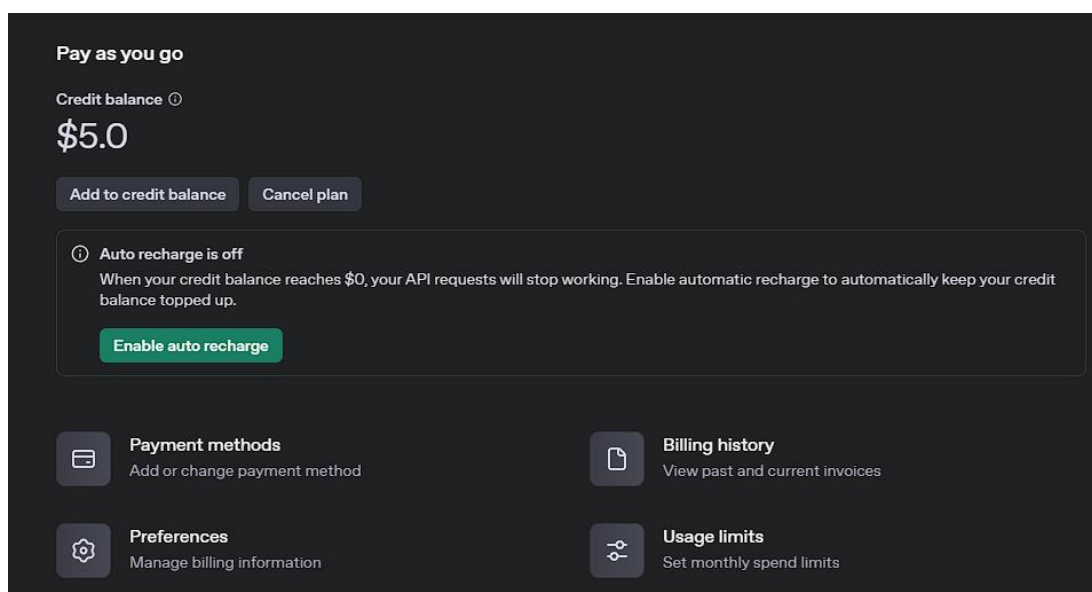


Рисунок 3.16 – Баланс моделі GPT

Попри це, важливо підкреслити, що для цілей даного експерименту жодна сума з балансу списана не була. Після внесення першого депозиту система активувала початкові безкоштовні токени, яких виявилось

достатньо для обробки обмеженої кількості запитів, необхідної для дослідницьких цілей. Таким чином, поповнення балансу було скоріше адміністративною вимогою, ніж фактичним розрахунком за використання.

Така політика доступу принципово відрізняє GPT від моделей BLOOM чи LLaMA, які можна використовувати локально та без жодних фінансових умов. GPT, у свою чергу, є комерційним продуктом, що функціонує як програмна послуга (API-as-a-Service), і навіть у випадках надання безкоштовних лімітів, користувач зобов'язаний пройти повний цикл верифікації.

У результаті, з одного боку, GPT демонструє високу якість генерації та стабільність обробки запитів, однак з іншого – вимагає суворої прив'язки до платіжної інфраструктури. У контексті даного дослідження це не створило технічних ускладнень, проте вимагає врахування таких обмежень у майбутніх дослідницьких чи продуктивних системах, які планують масштабне використання GPT через API.

У рамках підготовки до запуску експерименту з використанням моделі GPT через API OpenAI, критично важливим етапом стало коректне налаштування токена доступу. Токен (API-ключ) є унікальним ідентифікатором користувача, за допомогою якого OpenAI проводить авторизацію запитів, облік використаних ресурсів і контроль доступу до конкретних версій моделей, таких як GPT-3.5 або GPT-4 (рисунок 3.17). Цей ключ є конфіденційним і не повинен передаватися або зберігатися у відкритому вигляді в коді чи спільних файлах.



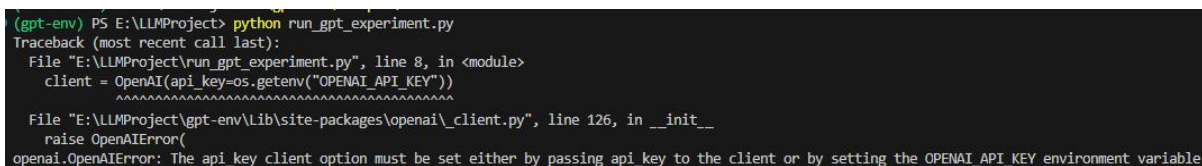
NAME	SECRET KEY	PROJECT ACCESS	PERMISSIONS
ForStudy	sk-...uX4A	Default project	All  

Рисунок 3.17 – Токен доступу GPT

У процесі реалізації експериментального скрипту було вирішено використовувати рекомендований спосіб передачі токена через змінну середовища, що є як безпечним, так і зручним підходом для одноразового налаштування сесії. Для цього в середовищі PowerShell (яке використовувалося як основна оболонка командного рядка у проєкті) токен вводився безпосередньо перед запуском основного скрипту за допомогою команди «`$env:OPENAI_API_KEY=«your-key-here»`». Така конструкція дозволяє призначити значення змінній середовища `OPENAI_API_KEY` лише в межах поточної сесії PowerShell. Це означає, що при завершенні сеансу або перезапуску терміналу змінна буде очищена автоматично, не залишаючи токен у системі. Це рішення не тільки спрощує роботу, але й відповідає базовим принципам безпеки при роботі з API.

Без коректного введення токена доступу, модель буде повертати помилку авторизації, як на рисунку 3.18.



```
(gpt-env) PS E:\LLMProject> python run_gpt_experiment.py
Traceback (most recent call last):
  File "E:\LLMProject\run_gpt_experiment.py", line 8, in <module>
    client = OpenAI(api_key=os.getenv("OPENAI_API_KEY"))
    ~~~~~
  File "E:\LLMProject\gpt-env\Lib\site-packages\openai\_client.py", line 126, in __init__
    raise OpenAIError(
openai.OpenAIError: The api_key client option must be set either by passing api_key to the client or by setting the OPENAI_API_KEY environment variable
```

Рисунок 3.18 – Помилка відсутності токена доступу

Сам скрипт звертається до цієї змінної за допомогою модуля `os.getenv(...)`, який дозволяє динамічно зчитувати значення без жорсткого кодування. Завдяки цьому забезпечується повна ізоляція ключа від основного коду, що робить реалізацію більш надійною й придатною до повторного використання.

Зазначений спосіб особливо зручний при роботі з кількома середовищами, коли необхідно швидко змінити API-ключ без внесення змін у кодову базу. Крім того, він легко адаптується для запуску скриптів у

середовищах CI/CD, серверних середовищах або під час автоматизованого тестування, де безпека й контроль за змінними середовища є стандартною практикою.

Таким чином, введення токена через `$env:OPENAI_API_KEY=«...»` виявилось простим і водночас ефективним способом інтеграції авторизаційного механізму в структуру проєкту, що забезпечив безпечний і стабільний доступ до GPT-моделі в межах експериментального запуску.

У процесі створення скрипту для запуску моделі GPT через OpenAI API було збережено загальну структуру, успішно застосовану під час роботи з моделями BLOOM та LLaMA, однак внесено низку адаптацій, пов'язаних зі специфікою віддаленого доступу до сервісу. На відміну від локального запуску моделей із бібліотеки Hugging Face Transformers, GPT не вимагає завантаження самої моделі на диск, але натомість потребує налаштування стабільного інтернет-з'єднання та коректної авторизації через API-ключ. Це дозволило значно скоротити обсяг локальних обчислень і вимоги до апаратного забезпечення.

Скрипт зберіг логіку попередньої реалізації – зчитування датасету Natural Questions, парсинг основних полів (питання, контекст, референсні короткі та довгі відповіді), генерацію запитів до мовної моделі та збереження результатів. Проте основна різниця полягала в механізмі генерації відповідей: у випадку GPT кожен запит надсилався до хмарного API-ендпоінту OpenAI, а не виконувався локально. В результаті було використано бібліотеку `openai`, яка дозволяє взаємодіяти з API простими методами, зокрема `openai.ChatCompletion.create(...)`.

Формат вхідного промпту також залишився уніфікованим для забезпечення коректності порівняльного аналізу, однак сам виклик моделі мав інший синтаксис. У GPT використовувалась система ролей (`system`, `user`, `assistant`), тож скрипт формував запит у вигляді списку словників, де роль `user` містила сформульоване питання разом із контекстом. Це

вимагало врахування особливостей побудови діалогового інтерфейсу GPT, відмінного від односторонньої генерації у BLOOM та LLaMA.

Ще однією важливою відмінністю було обмеження на довжину запиту, що задавалось через параметри `max_tokens`, `temperature`, `top_p` та інші. Вони дозволяли тонко налаштовувати поведінку генерації, що відсутнє у звичайних локальних моделях. У скрипті було обрано помірні значення для збереження балансу між якістю та стабільністю генерації.

Оскільки GPT працює віддалено, результати приходили у вигляді JSON-структури з вкладеним полем `choices[0].message['content']`, яке містило згенеровану відповідь. Цей текст було збережено разом із усіма іншими метаданими (питання, контекст, референс) у фінальний JSON-файл для подальшого оцінювання. Такий підхід забезпечив єдність структури вихідних даних з іншими моделями.

Таким чином, попри схожість логіки обробки запитань, скрипт для GPT принципово відрізнявся способом взаємодії з моделлю, наявністю мережевого шару, необхідністю авторизації та використанням спеціалізованої бібліотеки API. Це накладало нові вимоги до реалізації, але також відкривало переваги хмарного обчислення, зокрема швидкість відповіді та відсутність потреби в локальному ресурсі.

Під час оцінювання повноти відповідей, згенерованих моделлю GPT, було зафіксовано помітно вищий рівень деталізації у порівнянні з іншими протестованими моделями. В більшості випадків модель не обмежувалася короткими формулюваннями, а прагнула надавати розгорнуту інформацію, яка охоплювала всі релевантні аспекти запитання. Це стало особливо очевидним при роботі з питаннями, що мали складну структуру або вимагали залучення непрямих знань із наведеного контексту.

Модель впевнено оперувала різними рівнями абстракції, комбінуючи загальні факти з конкретними деталями, що сприяло підвищенню як інформативності відповідей, так і їх пояснювального потенціалу. У деяких випадках GPT додавала уточнення, пов'язані з можливими

інтерпретаціями запитання, або пояснювала, чому саме було обрано ту чи іншу відповідь. Це вказує на наявність високорівневої семантичної обробки запиту та контексту.

Також варто зауважити, що навіть при неповному або фрагментованому контексті модель намагалася компенсувати нестачу інформації за рахунок логічних припущень або узагальнених знань, притаманних її тренувальному корпусу. Такий підхід дозволив отримати відповіді, які залишалися інформативними навіть за умов обмежених вхідних даних. Проте варто розуміти, що така стратегія також може призвести до генерації некоректних фактів.

Аналіз повноти також показав, що GPT рідко уникає відповіді або залишає її пустою – на відміну від інших моделей, зокрема BLOOM або LLaMA, які у деяких випадках заявляли про відсутність достатньої інформації. GPT натомість демонструвала намагання запропонувати хоча б базовий варіант, навіть якщо впевненість у його точності могла бути невисокою. Така поведінка – наслідок її конструкції як інструменту діалогової генерації, який прагне бути корисним у будь-яких умовах.

Загалом, повнота відповідей GPT в межах проведеного експерименту була найвищою серед усіх моделей. Це дозволяє зробити попередній висновок про її придатність для завдань, де очікується не просто фактологічна відповідь, а комплексне пояснення, що враховує нюанси питання та контексту. У подальших етапах роботи буде доречно розглянути, як саме це впливає на загальну якість відповідей за іншими метриками.

Контекстна релевантність відповідей GPT виявилася значно вищою, ніж у попередньо протестованих моделей. Під час аналізу результатів було зафіксовано, що GPT у переважній більшості випадків точно відсилається до конкретної частини контексту, виявляючи здатність встановлювати логічні й семантичні зв'язки між запитанням і текстом. Модель демонструвала вміння оперувати як безпосередніми фактами, так і

опосередкованими зв'язками, що виходять за межі простого повторення знайденої інформації.

Відповіді, згенеровані GPT, не лише містили правильні фрагменти з контексту, а й надавали їх у зрозумілому для користувача форматі, адаптуючи мовні конструкції до формату відповіді. Наприклад, замість прямої цитати з тексту GPT переформулювала інформацію у формі розгорнутої пояснювальної відповіді. Це свідчить про глибший рівень розуміння семантики контексту, ніж той, що продемонстрували BLOOM або LLaMA.

Також слід зазначити, що навіть за наявності кількох потенційно релевантних частин у контексті, GPT часто вдавалося коректно ідентифікувати найбільш релевантний фрагмент для конкретного запиту. Це важливо для реалістичних сценаріїв застосування, де запитання може мати неоднозначну інтерпретацію або кілька потенційних відповідей. У таких випадках GPT проявляла здатність до пріоритезації інформації, що наближає її до поведінки експертної системи.

Ще одним важливим аспектом є стабільність релевантності: незалежно від складності чи довжини контексту, модель підтримувала відносно однаковий рівень точності у співставленні запитання з відповідною інформацією. Це дозволяє зробити припущення про сталість її механізмів пошуку значущого змісту в межах наданих даних. Для порівняння, BLOOM іноді вибирала фрагменти контексту довільно або повторювала вхідний текст без осмисленої переробки.

Попри це, були зафіксовані поодинокі випадки, коли GPT відхилялася від теми або використовувала другорядну інформацію замість основної. Проте такі ситуації траплялися рідко і, найчастіше, виникали за умов двозначного формулювання питання або надто об'ємного контексту. У таких випадках проблема полягала швидше в обмеженнях постановки задачі, ніж у слабкості самої моделі.

Таким чином, за параметром контекстної релевантності GPT продемонструвала високу точність, адаптивність і стійкість. Її відповіді залишалися логічно зв'язаними з контекстом, зберігаючи змістову цілісність та актуальність. Цей показник є критичним у практичному застосуванні моделей питання-відповіді, особливо в галузях, де важлива точність інтерпретації, зокрема в медицині, праві чи освіті.

Гнучкість у форматі відповіді є однією з найпомітніших переваг моделі GPT. На відміну від інших протестованих моделей, таких як BLOOM або LLaMA, GPT продемонструвала вміння динамічно адаптувати структуру відповіді до типу запитання, його складності та навіть до ймовірного наміру користувача. Наприклад, при простих запитаннях на зразок «What year was X founded?», GPT відповідала лаконічно і точно. У той же час, на складні запитання з кількома смисловими рівнями, вона формувала більш розгорнуті пояснення, з додатковим контекстом або обґрунтуванням.

Ця гнучкість особливо проявлялась у здатності моделі переключатися між різними стилями відповіді – від однослівних тверджень до декількох речень із прикладами. GPT змогла сформулювати відповідь у формі речення, що граматично й логічно завершувало запит, замість безпосереднього копіювання фрагменту контексту, як це часто робили інші моделі. Такий підхід дозволяє застосовувати GPT у сценаріях, що вимагають природного спілкування, а не лише механічного витягання даних.

Деякі запитання вимагали не просто повторення інформації, а й трансформації структури даних. Наприклад, якщо в контексті був перелік подій, GPT могла згрупувати їх у відповіді або подати у вигляді узагальнення. Це свідчить про здатність до когнітивної реконструкції даних, що суттєво перевершує можливості моделей, які опираються переважно на шаблонне генерування. Також модель коректно працювала з

різними типами відповідей – числовими, фактичними, описовими – і не змінювала формат, якщо цього не вимагала сама логіка запиту.

Ще одним важливим проявом гнучкості стало уникнення надмірного повторення фраз чи частин контексту, що було притаманне BLOOM. GPT формувала відповіді, які не дублювали повністю зміст запитання або контексту, а пропонували нову мовну конструкцію на основі наданих даних. Таким чином, модель не лише «відповідала», а й «переформулювала» думку, що властиво саме розвиненим мовним системам.

У випадках, коли відповідь не могла бути однозначною або контекст містив суперечливу інформацію, GPT не повертала довільний текст, а натомість намагалася обережно вказати на неоднозначність. Це може проявлятися в обережному формулюванні: «It is likely that...», «Sources suggest that...». Такий підхід свідчить про високий рівень мовного моделювання й логічного балансування у відповідях.

Однак у деяких випадках гнучкість формату виявлялась і як недолік: GPT іноді створювала занадто розгорнуті або стилістично «перевантажені» відповіді, навіть якщо питання потребувало простої відповіді. Це пов'язано з природою самої моделі, яка орієнтована на генерацію максимально релевантного тексту, що може бути й надмірним для конкретного випадку. Але загалом така поведінка скоріше вказує на універсальність, ніж на проблему.

З огляду на всі аспекти, GPT показала себе як найбільш гнучка модель серед протестованих. Її здатність варіювати довжину, стиль і структуру відповіді в залежності від запиту створює можливості для широкого використання в системах інтелектуального діалогу, освіти, автоматизованих помічників і дослідницьких систем.

Стійкість до неоднозначності – це ключовий показник здатності мовної моделі адекватно реагувати на запитання, в яких відсутня чітко визначена відповідь, або ж у контексті наявна суперечлива або неповна

інформація. У випадку GPT ця стійкість проявилася досить впевнено. Модель не лише намагалася вгадати правильну відповідь, а й часто демонструвала обережність у формулюваннях. У багатьох прикладах вона використовувала вирази на кшталт «There is no definitive answer», «According to some sources» або «It is likely that», що свідчить про вбудовану стратегію уникання категоричних тверджень у ситуаціях, де вони недоречні.

Важливою ознакою стійкості до неоднозначності є те, що GPT не виводила штучно згенеровані відповіді, засновані на припущеннях або халлюцинаціях. У випадках, коли відповідь не могла бути сформована на основі контексту, модель залишала запит без спекулятивних тверджень, що позитивно відрізняє її від деяких моделей, які «вгадували» або генерували фіктивні факти. Це дозволяє використовувати GPT у ситуаціях, що потребують довіри до джерел та коректної інтерпретації неоднозначних формулювань.

Ще однією формою прояву стійкості було вміння GPT адекватно інтерпретувати запитання з багатозначними поняттями. Наприклад, якщо у питанні була фраза «Who led the revolution?», а в контексті згадувалося декілька осіб з різними ролями, модель намагалася узагальнити або вказати на кілька можливих відповідей, пояснюючи різні варіанти. Така поведінка є ознакою когнітивної гнучкості – здатності працювати в умовах невизначеності, не знижуючи якість відповіді.

У певних випадках GPT також демонструвала обережне ставлення до дат, імен або подій, коли контекст подавав їх у неясному вигляді. Наприклад, якщо дата події була згадана у форматі «early 20th century», модель не конвертувала це у конкретний рік, як це робили інші моделі, а залишала приблизне формулювання, аби не створити хибне враження точності. Такий підхід важливий при роботі з навчальними, історичними або журналістськими текстами, де надмірна точність може бути викривленням змісту.

Проте слід зазначити, що у поодиноких випадках GPT могла все ж спробувати дати відповідь навіть у відсутності достатнього контексту. Це виявлялося рідко і зазвичай супроводжувалося обережними формулюваннями, однак подібна поведінка свідчить про обмеження моделі, які варто враховувати при її використанні у критичних системах.

Загалом, GPT продемонструвала високу стійкість до неоднозначності, що є критично важливим аспектом для її інтеграції в середовища, де модель має працювати не лише з точними фактами, а й із неповною або суперечливою інформацією. Ця властивість підвищує її придатність до використання у сфері права, медицини, освіти, наукових досліджень, де неоднозначність є нормою, а не винятком.

На основі проведеного експерименту з використанням моделі GPT можна зробити низку обґрунтованих висновків щодо її здатності ефективно виконувати завдання питання-відповіді в межах обраного датасету. В першу чергу слід відзначити, що модель демонструє високу стабільність у генерації змістовних, логічно завершених і граматично коректних відповідей. Незалежно від складності або структури запитання, GPT здебільшого формує відповідь, яка не лише формально задовольняє умови завдання, а й зберігає цілісність смислового навантаження. Це свідчить про глибоку інтеграцію мовних закономірностей у її архітектуру.

Особливу увагу заслуговує здатність моделі до адаптивної побудови відповіді залежно від стилю та змісту контексту. GPT, на відміну від деяких інших моделей, не схильна до формального підходу, а намагається вбудовуватися в логіку викладу інформації, що вже надана. У багатьох випадках вона відтворювала ритм, термінологію або навіть інтонаційні особливості запитання, що створювало враження природної взаємодії та зменшувало дисонанс між вхідними й вихідними даними.

Також варто зауважити, що модель доволі впевнено працює з різними форматами відповідей. Вона однаково ефективно генерує як однофразові резюме, так і розгорнуті пояснення, якщо цього вимагає

контекст. Така гнучкість є надзвичайно цінною у практичному застосуванні, оскільки дозволяє адаптувати вихід моделі до очікувань користувача без жорсткого регулювання. GPT не демонструє сліпого дотримання шаблонів, а виявляє здатність до ситуаційного аналізу і побудови релевантної відповіді на основі змісту.

Ще одним важливим фактором є її стійкість до неоднозначності. GPT не уникає складних запитань, а натомість намагається надати відповідь із застереженням, або пояснити межі достовірності твердження. Така поведінка демонструє наявність внутрішніх механізмів оцінки впевненості, що дозволяє уникати «галюцинацій» – тобто випадкових або вигаданих фактів. У цьому аспекті модель вигідно вирізняється своєю здатністю утримуватися від неперевірених припущень, що є особливо критичним для застосування в контекстах, де важлива надійність.

Загалом, результати свідчать про високу якість реалізації архітектури GPT у сфері завдань питання-відповіді. Вона поєднує точність, гнучкість, обережність і здатність працювати з різноманітними форматами, що робить її придатною до широкого спектра завдань – від навчальних і дослідницьких до професійно-інформаційних. Водночас отримані характеристики також відкривають перспективи подальшої адаптації моделі під специфічні задачі, включаючи доменно-орієнтоване навчання або інтеграцію в гібридні системи підтримки рішень.

3.2 Оцінювання відповідей моделей згідно обраних метрик

3.2.1 Створення скрипту оцінювання відповідей

У рамках проведення повноцінного порівняльного аналізу відповідей, згенерованих трьома великими мовними моделями – BLOOMZ, LLaMA3 та GPT – було реалізовано спеціалізований скрипт для автоматичного обчислення метрик якості. Автоматизоване оцінювання дає змогу

об'єктивно та систематично зіставити точність, повноту та релевантність відповідей кожної з моделей на однаковому наборі запитань.

Основна мета цього етапу полягає у переведенні якісного враження від роботи моделей у формалізовані числові показники. Це дозволяє не лише порівняти між собою результати, але й чітко виявити переваги або слабкі сторони кожної з моделей. Для оцінки обрано низку відомих метрик з галузі обробки природної мови, що дозволяють аналізувати подібність згенерованих відповідей до еталонних з різних точок зору.

Задля забезпечення репрезентативності результатів, скрипт опрацьовує всі збережені результати моделей, отримані на основі єдиного датасету, використовуючи єдину систему координат для порівняння – обрані метрики. Реалізація скрипту забезпечує масштабованість, можливість розширення і повторне використання для інших наборів даних або моделей у майбутньому.

Під час реалізації автоматизованого оцінювання якості відповідей, згенерованих великими мовними моделями, було використано низку бібліотек Python, які є загальноприйнятими інструментами в сфері обробки природної мови (NLP). Їхнє використання дозволило реалізувати порівняльний аналіз із залученням як класичних, так і сучасних метрик подібності текстів, що базуються як на лексичному збігу, так і на семантичному узгодженні.

Однією з основних бібліотек стала nltk (Natural Language Toolkit), яка була використана для розрахунку метрики BLEU. Ця метрика дозволяє оцінити точність відповідей на основі збігу n-грам між гіпотезою (відповіддю моделі) та еталонною відповіддю. У рамках цього проєкту використовувався метод `sentence_bleu` з реалізацією згладжування результатів, що забезпечує більш стабільну поведінку при коротких відповідях.

Для розрахунку метрик ROUGE-1 та ROUGE-L було залучено бібліотеку `rouge_score`, розроблену Google Research. Вона дозволяє

обчислювати збіг за окремими словами, а також враховує довгі спільні підрядки (Longest Common Subsequence) між відповідями. Завдяки використанню морфологічного стемінгу та гнучких правил підрахунку, ця бібліотека є ефективним інструментом для лексичного аналізу якості генеративних моделей.

Особливе місце посіла бібліотека `bert_score`, яка дозволяє оцінювати подібність відповідей на основі глибоких контекстуальних представлень, сформованих трансформерними моделями. В основі метрики BERTScore лежить ідея порівняння векторів слів, згенерованих попередньо натренованими мовними моделями, зокрема RoBERTa. Завдяки цьому вдалося врахувати семантичну близькість слів навіть за відсутності прямого лексичного збігу.

Для роботи з вхідними та вихідними файлами, обробки результатів і збереження підсумкових таблиць була використана бібліотека `pandas`. Вона дозволила реалізувати обробку структурованих даних у вигляді таблиць, легко зчитувати JSON-формати, а також зберігати результати оцінювання у форматі CSV для подальшого аналізу.

Комплексне використання зазначених бібліотек надало змогу створити гнучкий, масштабований і повторюваний інструмент для оцінки моделей. Завдяки їм було забезпечено баланс між простотою реалізації, якістю обчислень і можливістю розширення під час подальших експериментів.

Код скрипту порівняння наведено на лістингу 3.1.

Лістинг 3.1 – Скрипт порівняння метрик

```
import json
import pandas as pd
from nltk.translate.bleu_score import sentence_bleu,
SmoothingFunction
from rouge_score import rouge_scorer
from bert_score import score as bert_score
files = {
    «bloomz»: «bloomz_results.json»,
    «llama»: «llama3_results.json»,
```

Продовження лістингу 3.1

```

    «gpt»: «gpt_results.json»
}
records = []
rouge = rouge_scorer.RougeScorer(['rouge1', 'rougeL'],
use_stemmer=True)
smoothie = SmoothingFunction().method4

for model_name, path in files.items():
    with open(path, «r», encoding=«utf-8») as f:
        data = json.load(f)

    for item in data:
        reference = item.get(«reference_long», «»)
        prediction = item.get(f»{model_name}_answer», «»)
        question = item.get(«question», «»)
        if not reference.strip() or not prediction.strip():
            continue

        bleu = sentence_bleu([reference.split()],
prediction.split(), smoothing_function=smoothie)
        rouge_scores = rouge.score(reference, prediction)
        rouge1 = rouge_scores[«rouge1»].fmeasure
        rougeL = rouge_scores[«rougeL»].fmeasure
        P, R, F1 = bert_score([prediction], [reference],
lang=«en», verbose=False)
        bert_f1 = F1[0].item()

        records.append({
            «model»: model_name,
            «question»: question,
            «reference»: reference,
            «prediction»: prediction,
            «bleu»: bleu,
            «rouge1»: rouge1,
            «rougeL»: rougeL,
            «bertscore_f1»: bert_f1
        })
df = pd.DataFrame(records)
df.to_csv(«evaluation_results.csv», index=False,
encoding=«utf-8»)
print(«✓ Оцінювання завершено. Результати збережено у
evaluation_results.csv»)

```

Поданий скрипт реалізує автоматичне оцінювання відповідей трьох великих мовних моделей – BLOOMZ, LLaMA3 та GPT – за допомогою метрик BLEU, ROUGE-1, ROUGE-L та BERTScore.

На першому етапі скрипт зчитує дані з трьох JSON-файлів, у кожному з яких збережено питання, еталонну відповідь (`reference_long`) і відповідь певної моделі. Далі для кожної пари «еталон – відповідь моделі» обчислюються метрики: BLEU (на основі n-грам), ROUGE (на основі точного збігу слів і довгих підрядків) та BERTScore (семантична подібність за допомогою глибоких векторних представлень).

Результати кожної оцінки зберігаються у вигляді таблиці, яка експортується у файл `evaluation_results.csv` для подальшого аналізу. Скрипт є гнучким і легко масштабованим для оцінки нових моделей або датасетів.

Після успішного виконання попереднього етапу оцінювання відповідей великих мовних моделей було прийнято рішення оптимізувати структуру результатів з метою подальшого аналізу й візуалізації. З огляду на це, було створено окремий скрипт, завданням якого стало видалення надлишкової інформації, що не є релевантною для математичної обробки результатів – зокрема, текстів самих запитань, відповідей моделей, а також референсних відповідей.

Подібна трансформація дозволила отримати компактну й придатну для обчислень таблицю, яка містить лише суттєві числові значення: назву моделі, значення BLEU, ROUGE-1, ROUGE-L та BERTScore для кожної відповіді. Зменшення обсягу структурованих даних сприяло як зниженню навантаження на засоби візуалізації, так і покращенню зручності інтеграції цих результатів у аналітичні модулі, що будуть описані далі.

Формування скороченого представлення даних було реалізовано за допомогою скрипту, що здійснює зчитування повного результату оцінювання з CSV-файлу, обирає необхідні колонки та зберігає новий файл, що містить лише назву моделі та відповідні числові оцінки. Таким чином, було створено основу для подальшого порівняльного аналізу ефективності моделей у рамках обраних метрик. Код скрипту наведено на лістингу 3.2.

Лістинг 3.2 – Скрипт відокремлення результатів

```
import pandas as pd
df = pd.read_csv(«evaluation_results.csv», encoding=«utf-8»)
metrics_df = df[['model», «bleu», «rouge1», «rougeL», «bertscore_f1»]]
metrics_df.to_csv(«evaluation_metrics_only.csv», index=False, encoding=«utf-8»)
```

Цей скрипт завантажує CSV-файл з повними результатами оцінювання відповідей мовних моделей, обирає лише ключові метрики (BLEU, ROUGE-1, ROUGE-L, BERTScore) разом із назвою моделі та зберігає скорочену версію у новий файл `evaluation_metrics_only.csv`. Це дозволяє спростити подальший аналіз і візуалізацію результатів.

На наступному етапі аналізу було реалізовано скрипт для візуалізації результатів оцінювання відповідей моделей на основі згенерованого файлу `evaluation_metrics_only.csv`. Метою цього етапу є забезпечення зручного та наочного порівняння якості відповідей моделей BLOOMZ, LLaMA та GPT відповідно до чотирьох основних метрик: BLEU, ROUGE-1, ROUGE-L та BERTScore.

Візуалізація проводиться у вигляді стовпчикових діаграм для кожної окремої метрики. Для кожної з них обчислюється середнє значення по всіх відповідях кожної моделі. Отримані середні значення наочно демонструють, яка з моделей має вищу узагальнену якість відповіді згідно з відповідною метрикою. Це дозволяє швидко оцінити сильні та слабкі сторони кожної з моделей у контексті задачі генерації відповідей на запитання.

Скрипт було реалізовано з використанням бібліотеки `matplotlib`, що дозволяє будувати графіки без зайвих стилістичних обмежень, а також надає гнучкість у налаштуванні візуальних елементів, таких як назви осей, підписи стовпців, підписи моделей та масштабування. Кожна метрика виводиться окремо, що дозволяє фокусувати увагу на конкретному аспекті якості моделей. Скрипт наведено на лістингу 3.3.

Лістинг 3.3 – Скрипт створення графіків оцінок.

```

import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv(«evaluation_metrics_only.csv»,
encoding=«utf-8»)
mean_scores = df.groupby(«model»)[[«bleu», «rouge1»,
«rougeL», «bertscore_f1»]].mean().reset_index()
metrics = [«bleu», «rouge1», «rougeL», «bertscore_f1»]
titles = [«BLEU Score», «ROUGE-1 Score», «ROUGE-L Score»,
«BERTScore (F1)»]

for metric, title in zip(metrics, titles):
    plt.figure(figsize=(6, 4))
    plt.bar(mean_scores[«model»], mean_scores[metric],
color='skyblue')
    plt.title(title)
    plt.ylabel(«Score»)
    plt.grid(axis='y', linestyle='--', alpha=0.6)
    plt.tight_layout()
    plt.show()

```

Результати графічного відображення значно спрощують процес інтерпретації оцінок, оскільки візуальне порівняння дозволяє швидко виявити відмінності між моделями та зробити попередні висновки про загальну ефективність кожної з них.

Наведений скрипт виконує повний цикл побудови стовпчикових діаграм для візуального порівняння якості відповідей трьох мовних моделей (BLOOMZ, LLaMA та GPT) за чотирма метриками: BLEU, ROUGE-1, ROUGE-L та BERTScore (F1). Для цього використовується бібліотека pandas для роботи з табличними даними та matplotlib.pyplot для побудови графіків.

На першому етапі зчитується CSV-файл evaluation_metrics_only.csv, який містить оцінки для кожного запитання. Далі виконується групування даних за назвою моделі та обчислення середнього значення кожної з чотирьох метрик. Таким чином, формується таблиця з усередненими результатами, що дозволяє оцінити загальну ефективність кожної моделі.

Для кожної метрики по черзі створюється окремий графік. В межах циклу для кожного з показників генерується стовпчикова діаграма, де по осі X розміщуються назви моделей, а по осі Y – середнє значення метрики.

Діапазон значень по осі Y обмежено інтервалом $[0, 1]$, що є стандартною шкалою для таких метрик. Графік оформлено з використанням сітки для кращої візуальної сприйнятливості та збережено як зображення у форматі PNG з назвою, відповідною метриці (наприклад, `bleu_bar_chart.png`).

Таким чином, скрипт забезпечує автоматизований, зручний і наочний спосіб аналізу якості відповідей моделей згідно з основними метриками оцінювання.

Проведене оцінювання відповідей моделей дозволило здійснити кількісний аналіз якості згенерованих текстів за допомогою низки об'єктивних метрик – BLEU, ROUGE-1, ROUGE-L та BERTScore (F1). Розроблені скрипти автоматично обробили результати роботи моделей BLOOMZ, LLaMA та GPT, виділили основні показники та зберегли їх у зручному форматі для подальшого аналізу.

Застосування графічної візуалізації значно спростило порівняння ефективності моделей між собою. Побудовані діаграми надали наочне уявлення про сильні та слабкі сторони кожної моделі, дозволивши швидко ідентифікувати відхилення чи переваги.

3.2.2 Результати оцінювання відповідей

Нижче наведено результати оцінювання якості відповідей, згенерованих трьома мовними моделями – BLOOMZ-3B, LLaMA-3 та GPT-3.5 – на основі попередньо сформованої вибірки запитань з датасету Natural Questions. Для кожної відповіді було проведено обчислення ключових текстових метрик, які дозволяють кількісно оцінити точність, релевантність та схожість генерованих фрагментів із еталонними.

У якості метрик було використано BLEU, ROUGE-1, ROUGE-L та BERTScore (F1). Вони охоплюють різні аспекти відповідності – від точного збігу слів (BLEU), до наявності важливих підрядків (ROUGE) та глибшого семантичного збігу (BERTScore). Таке комплексне оцінювання

дозволяє врахувати як формальні, так і змістові аспекти генерованих відповідей, забезпечуючи глибший рівень порівняння моделей.

Результати метрик для кожної моделі були агреговані та представлені як у табличному, так і в графічному вигляді. Середні значення кожної метрики розраховано на основі всіх оброблених запитань. Це дозволяє зробити репрезентативний висновок щодо загального рівня ефективності кожної з моделей у розв'язанні завдань типу «питання-відповідь» на основі реального контексту.

У результаті виконаного автоматизованого оцінювання було сформовано зведений файл формату CSV, який містить детальні метрики якості для кожної відповіді, згенерованої мовними моделями. У цьому файлі кожен рядок відповідає окремому прикладу з тестової вибірки та містить оцінки за чотирма метриками: BLEU, ROUGE-1, ROUGE-L і BERTScore (F1), а також ідентифікатор моделі, яка сформувала відповідь.

Такий формат структурування даних дозволяє не лише проводити візуальний аналіз, а й виконувати математичну обробку результатів – зокрема, розрахунок середніх значень, порівняння моделей за кожною окремою метрикою, побудову графіків та формування статистичних висновків. Зразок фрагменту цього результативного файлу наведено у таблиці 3.1.

Таблиця 3.1 – Частина результатів оцінювання запитів

model	bleu	rouge1	rougeL	bertscore_f1
bloomz	2,9136E-257	0,002014099	0,002014099	0,738805115
bloomz	3,57227E-67	0,015625	0,015625	0,766141176
bloomz	0	0	0	0,73076576
bloomz	2,28569E-49	0,02173913	0,02173913	0,763357878
bloomz	6,3261E-59	0,01793722	0,01793722	0,785530806
bloomz	0	0	0	0,782894671
bloomz	6,6325E-111	0,008264463	0,008264463	0,714726985
bloomz	5,14111E-13	0,020408163	0,020408163	0,814599037
bloomz	1,14581E-17	0,057142857	0,057142857	0,825992346

Продовженні таблиці 3.1

bloomz	0	0,021505376	0,021505376	0,783243537
bloomz	0	0	0	0,76720041
bloomz	0	0	0	0,774376631
bloomz	1,97926E-32	0,032786885	0,032786885	0,75020504
bloomz	4,16589E-09	0,03030303	0,03030303	0,759329498
bloomz	0	0	0	0,744326472
bloomz	1,6063E-109	0,008948546	0,008948546	0,769181669
bloomz	0	0	0	0,710909069
bloomz	0	0,04040404	0,04040404	0,793711305
bloomz	3,11824E-13	0,02	0,02	0,772013724
bloomz	0	0	0	0,764645875
bloomz	0	0	0	0,706438959
bloomz	0	0	0	0,765800118

На рисунку представлено фрагмент результативної таблиці, сформованої за підсумками автоматизованого оцінювання якості відповідей, згенерованих моделлю BLOOMZ. Таблиця містить п'ять основних стовпців: model, bleu, rouge1, rougeL та bertscore_f1. У стовпці model зазначено назву моделі, яка відповідала на запитання. Інші стовпці містять числові значення відповідних метрик якості, обчислених для кожної конкретної відповіді.

Значення метрик наведено у форматі з фіксованою або експоненційною точністю, що є типовим для малих чисел. Низка рядків містить нульові значення – це вказує на випадки, коли відповідь моделі не мала жодного перекриття з еталонною відповіддю або коли була повністю відсутня. Ці результати дозволяють здійснити подальший порівняльний аналіз, статистичне узагальнення та виявлення закономірностей у поведінці моделей під час генерації відповідей.

На основі результатів, отриманих у табличному форматі, було проведено побудову візуальних графіків, що наочно ілюструють порівняння якості відповідей між трьома мовними моделями: BLOOMZ,

LLaMA та GPT. Кожна метрика оцінювання була подана окремим графіком, що відображає усереднене значення відповідної метрики для кожної з моделей. Це дозволяє швидко ідентифікувати загальні тенденції та виявити сильні чи слабкі сторони кожної архітектури в контексті завдання питання-відповіді.

Зокрема, на рисунку 3.19 представлено стовпчикову діаграму `bertscore_f1_bar_chart`, яка демонструє середні значення метрики BERTScore (F1) для кожної моделі. Ця метрика оцінює семантичну схожість між відповіддю моделі та еталонною відповіддю, враховуючи контекстні представлення слів. Діаграма наочно показує, наскільки добре моделі оперують глибоким змістом при побудові відповідей.

Рисунок 3.20 – `bleu_bar_chart` – відображає результати BLEU, що базується на точному збігові фраз та слів між відповіддю та референсом. Цей графік дозволяє оцінити, наскільки дослівно модель відтворює правильну відповідь, що особливо корисно при аналізі точних фактологічних запитань.

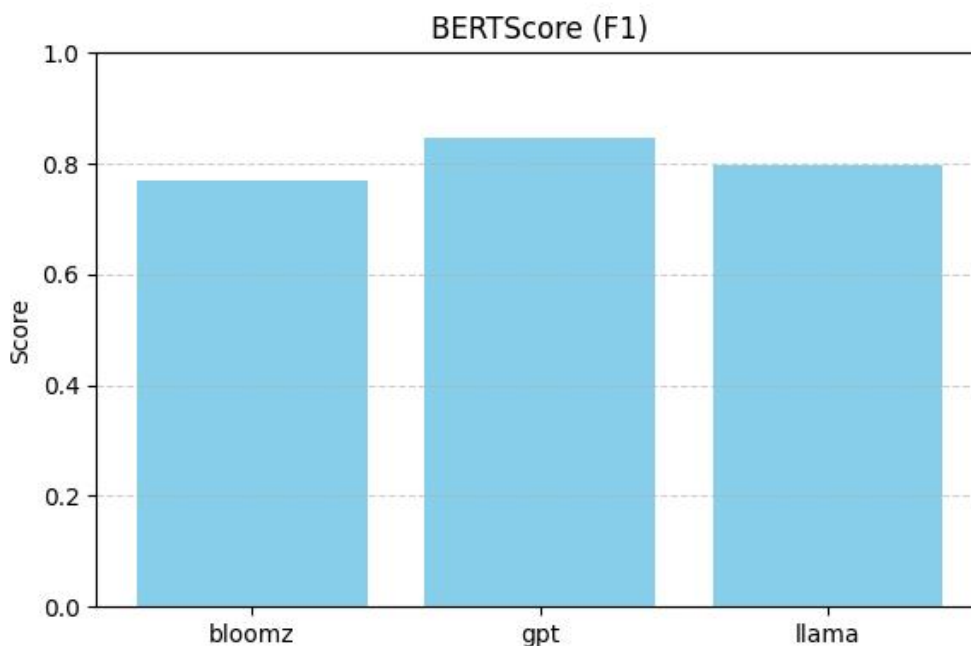


Рисунок 3.19 – Графік оцінки BestScore (F1)

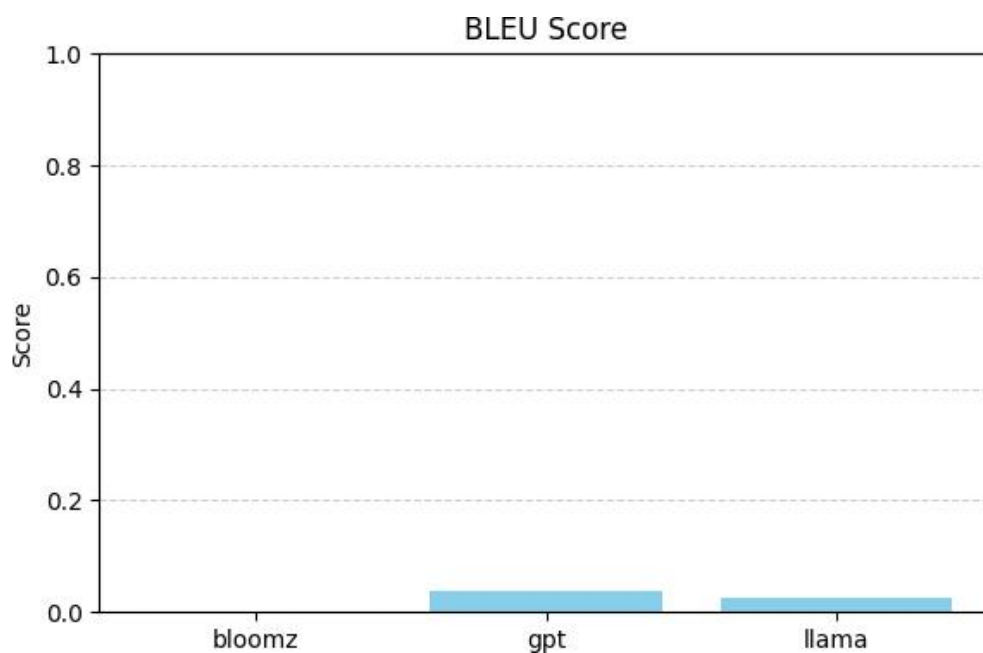


Рисунок 3.20 – Графік оцінки BLUE Score

На рисунку 3.21 – rouge1_bar_chart – зображено середні значення метрики ROUGE-1, яка враховує точне співпадіння окремих слів. Це дозволяє оцінити загальну лексичну відповідність тексту відповіді з еталоном.

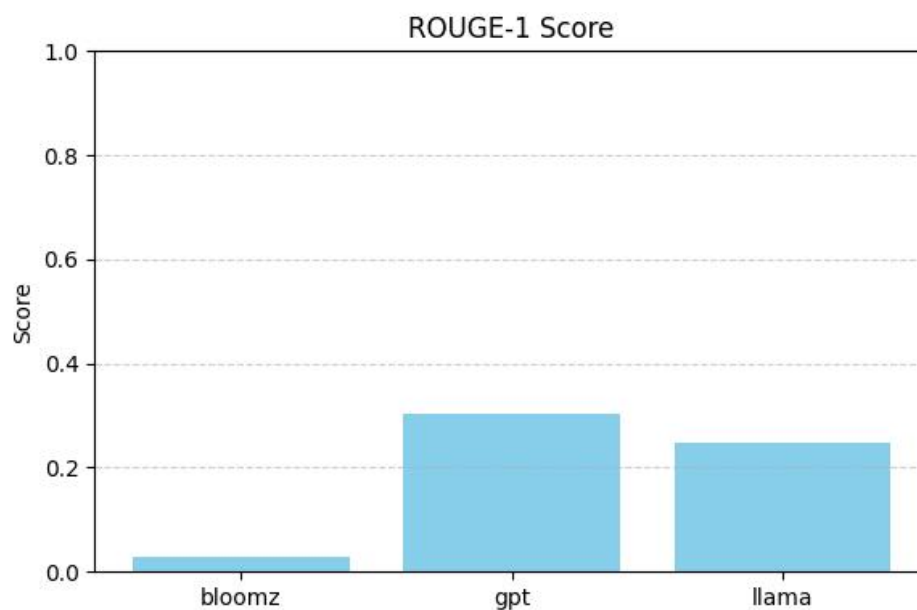


Рисунок 3.21 – Графік оцінки ROUGE-1 Score

Останній графік – рисунок 3.22, rougeL_bar_chart – ілюструє ROUGE-L, що базується на довжині найдовшої спільної підпоследовності (Longest Common Subsequence), що відображає структурну схожість відповідей.

Сукупність цих чотирьох візуалізацій забезпечує комплексне уявлення про поведінку кожної з моделей у різних вимірах якості тексту, що є надзвичайно важливим для обґрунтованого аналізу результатів експерименту.

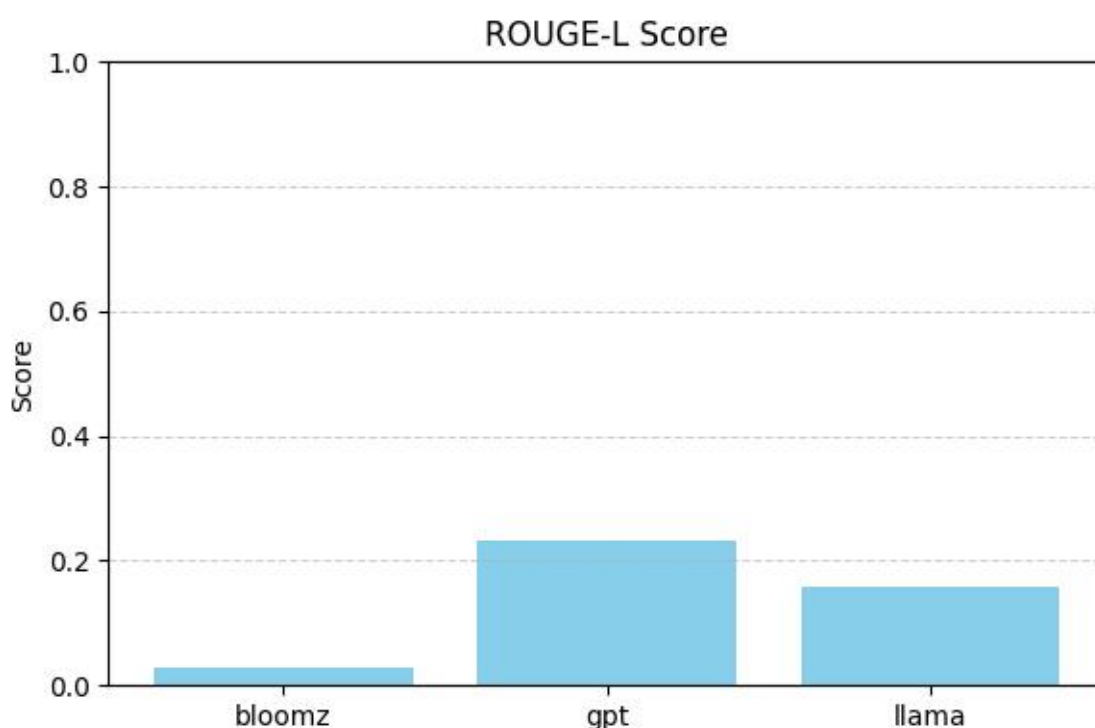


Рисунок 3.22 – Графік оцінки ROGUE-L Score

У результаті проведеного автоматизованого оцінювання було отримано повну картину якості відповідей, згенерованих кожною з моделей. Зведені числові значення метрик дозволили здійснити об'єктивне порівняння продуктивності моделей у задачі генерації відповідей на запитання. Побудовані візуалізації лише підтвердили спостережувані відмінності між підходами та продемонстрували стабільність або слабкі

місця кожної архітектури. Зібрані дані сформували основу для подальшого якісного аналізу, що буде представлено в наступному підрозділі.

3.2.3 Опис та аналіз результатів оцінювання

У цьому підрозділі представлено опис та аналітичне узагальнення результатів, отриманих під час автоматизованого оцінювання відповідей трьох великих мовних моделей. Проаналізовано поведінку кожної моделі на основі метрик BLEU, ROUGE-1, ROUGE-L та BERTScore (F1), що дозволяє здійснити порівняння не лише з точки зору точності відтворення еталонних фраз, але й у контексті семантичної відповідності та лексичної релевантності.

На основі агрегованих показників було виявлено відмінності у якості результатів, які на пряму залежать як від архітектурних особливостей моделей, так і від їхніх масштабів та навчальних стратегій. Оцінювання охоплювало різні рівні складності запитань, що дозволило протестувати моделі на реалістичних сценаріях, притаманних завданням типу «питання-відповідь».

Результат оцінки BestScore (F1) було наведено на рисунку 3.19.

На наведеній діаграмі відображено усереднені значення метрики BERTScore (F1) для трьох великих мовних моделей – BLOOMZ, LLaMA та GPT. Ця метрика є однією з найважливіших при оцінюванні задач на кшталт «питання-відповідь», оскільки вона враховує семантичну подібність між згенерованою відповіддю та еталонною. На відміну від метрик, що базуються на збігу слів або фраз, BERTScore порівнює контекстні векторні представлення слів, що дозволяє враховувати синонімію, перестановки та узгодженість змісту.

Згідно з результатами, найвищий показник було досягнуто моделлю GPT, середнє значення її BERTScore F1 перевищує 0.85. Це свідчить про те, що GPT найкраще передає зміст еталонної відповіді, навіть якщо лексично

її формулювання може відрізнятись. Такий результат є очікуваним з огляду на масштабність цієї моделі, її інструкційне донавчання, а також широке охоплення корпусу даних у процесі тренування.

Модель LLaMA показала дещо нижчий, але також конкурентний результат – близько 0.80. Це свідчить про достатньо високий рівень семантичної точності, хоча й зі спостережуваним відставанням у порівнянні з GPT. Варто зазначити, що навіть попри значні апаратні вимоги і високу точність генерації, LLaMA не завжди здатна до узагальнень на рівні глибоких семантичних структур, що може бути пов'язано з меншою гнучкістю промпт-орієнтованої генерації в її архітектурі.

Найнижчий результат зафіксовано у моделі BLOOMZ. Її середній показник не перевищує 0.78, що, зважаючи на масштаб моделі, є типовим. Такий рівень BERTScore свідчить про обмежену здатність моделі повноцінно враховувати змістовий контекст, що, ймовірно, пов'язано із менш ефективною оптимізацією на інструкційному навчанні або меншою ємністю архітектури в порівнянні з конкурентами.

Водночас варто зауважити, що всі три моделі продемонстрували загалом високі значення BERTScore, що свідчить про відносно якісний рівень генерації навіть у найпростішій конфігурації. Значення, близькі до 0.8, загалом вважаються прийнятними у завданнях генерації природної мови, особливо якщо питання мають відкритий або описовий характер.

Перевага GPT за цією метрикою вказує на її здатність точніше узгоджувати логічні й семантичні залежності у відповіді, навіть за умов обмеженого контексту. Це є важливою характеристикою для прикладних систем, де значення має не лише формальний збіг, а й правильність переданого змісту.

Отримані результати підтверджують, що BERTScore є цінним інструментом оцінювання саме в тих випадках, коли відповіді можуть мати різні формулювання, але однакове смислове навантаження. У такому

контексті GPT демонструє найкращі результати, тоді як BLOOMZ потребує додаткової оптимізації або масштабування для досягнення порівняної якості.

Результат оцінки BLUE Score було наведено на рисунку 3.20.

На діаграмі, що відображає середні значення BLEU-метрики для кожної з досліджуваних моделей, чітко помітно, що всі три системи демонструють низький рівень показників за цією метрикою. Значення не перевищують навіть позначки у 0.05, що є індикатором незначного збігу з референсною відповіддю на рівні точного порядку слів. Це є очікуваним, з огляду на специфіку BLEU – вона є метрикою, орієнтованою на точне повторення фраз, що недостатньо добре працює в умовах вільної генерації мовного тексту, особливо коли правильна відповідь може бути сформульована різними способами.

Модель GPT, попри низький абсолютний результат, виявилася найкращою серед трьох моделей за цією метрикою. Це вказує на її здатність іноді формувати відповіді, близькі до тексту референсу не лише за змістом, а й за лексикою та порядком слів. Зважаючи на архітектурну складність і масштаб інструкційного донавчання GPT, її відносно вищий BLEU показник є додатковим свідченням точнішого узгодження лексичної форми відповіді з еталонною.

Результат LLaMA виявився нижчим за GPT, хоча й не значно. Вона іноді повторювала ключові слова з еталону, але рідше дотримувалася точного порядку слів або повної фразової структури, що є критичним для BLEU. Проте такі показники не обов'язково свідчать про низьку якість відповіді, а скоріше про відмінності у підходах до генерації: LLaMA більш вільно трансформує зміст, що не завжди призводить до збігів на рівні н-грам.

Найгірші показники продемонструвала модель BLOOMZ. Значення її BLEU-метрики фактично наближаються до нуля, що свідчить про майже повну відсутність лексичного збігу з еталонними відповідями. Такий

результат цілком узгоджується з попередніми спостереженнями щодо коротких, контекстно обмежених відповідей цієї моделі. BLEU дуже чутливий до довжини відповіді, і саме через це короткі відповіді BLOOMZ не отримали належного значення, навіть якщо вони частково відповідали на запитання.

Узагальнюючи, слід зазначити, що BLEU у контексті відкритих генеративних моделей для завдання «питання-відповідь» має обмежену інформативність. Його низькі значення не завжди відображають реальну якість відповіді, особливо коли моделі генерують переформульовану, але правильну відповідь. Метрика є більш корисною у задачах машинного перекладу або тоді, коли від моделі очікується дослівне відтворення тексту.

Разом з тим, порівняння BLEU між моделями надає певне уявлення про схильність кожної з них до дослівного повторення фраз з еталонів. GPT, з огляду на отримані результати, має перевагу у цьому аспекті, хоча загальна оцінка моделі повинна базуватися на ширшому спектрі метрик, які враховують семантику. Відповідно, у практичних сценаріях BLEU варто інтерпретувати обережно, лише в поєднанні з іншими індикаторами.

Результат оцінки ROUGE-1 Score було наведено на рисунку 3.21.

На наведеній діаграмі зображено результати оцінювання трьох моделей за метрикою ROUGE-1, яка фокусується на співпадінні окремих слів між згенерованою відповіддю та референсною. Ця метрика дозволяє оцінити здатність моделі зберігати ключові терміни у відповіді, навіть якщо порядок слів чи структура речення змінені. У контексті завдань з питань та відповідей, де важливо зберігати смислові одиниці, ROUGE-1 є більш гнучким індикатором порівняно з BLEU.

Результати демонструють суттєву перевагу моделі GPT, яка досягла найвищого значення серед усіх трьох систем. Це свідчить про те, що ця модель найчастіше використовувала ті ж слова, що й референс, що є свідченням її точного розуміння контексту та відповідного підбору термінів. Високий ROUGE-1 бал у GPT також узгоджується з попередніми

якісними спостереженнями, де ця модель формувала не лише релевантні, але й лексично наближені відповіді.

LLaMA посіла друге місце, хоча її показник ROUGE-1 дещо нижчий за GPT. Тим не менш, результат можна вважати прийнятним, адже LLaMA зберігає здатність відтворювати ключову лексику, хоча й не з тією ж точністю, що й GPT. Така різниця частково пояснюється особливостями генерації: LLaMA схильна до перефразування і може змінювати формулювання, що знижує метрику ROUGE-1, не знижуючи при цьому фактичну якість відповіді.

Модель BLOOMZ показала значно гірші результати. Її оцінка ROUGE-1 майже вдесятеро нижча, ніж у LLaMA, і майже повністю наближається до нуля. Це свідчить про майже повну відсутність спільної лексики з еталонною відповіддю у відповідях цієї моделі. Як зазначалося раніше, BLOOMZ часто генерує дуже короткі або поверхневі відповіді, що в багатьох випадках не включають ключові слова запиту або очікуваної відповіді.

З огляду на контекст, ROUGE-1 є більш інформативною метрикою, ніж BLEU, оскільки дозволяє виявити збіг навіть у випадку переставленого або скороченого речення. Ця гнучкість є важливою перевагою у оцінюванні генеративних відповідей. Той факт, що GPT значно випереджає інші моделі за цією метрикою, лише підсилює висновок про її загальну перевагу у здатності до відтворення очікуваної інформації.

Варто зазначити, що навіть порівняно високі значення ROUGE-1 не завжди гарантують правильну відповідь, проте вони є хорошим індикатором того, що модель «розуміє», про що йдеться, і використовує відповідну лексику. Саме це робить ROUGE-1 важливою складовою у комплексному аналізі якості.

Загалом, результати ROUGE-1 підтверджують висновки, зроблені при якісному аналізі: GPT демонструє найвищу релевантність відповідей, LLaMA – стабільну, хоча й слабшу поведінку, а BLOOMZ – значні

обмеження у формуванні відповідей, що відображається в найгіршому показнику серед трьох моделей.

Результат оцінки ROGUE-L Score було наведено на рисунку 3.22.

Метрика ROGUE-L оцінює схожість між згенерованою відповіддю та еталонною на основі довшої спільної підпоследовності слів. Це дозволяє враховувати не лише присутність ключових слів, як у ROGUE-1, але й їхній порядок та структурну відповідність. Такий підхід забезпечує більш глибоке розуміння синтаксичної близькості між відповідями, що є особливо важливим у випадках, коли правильна відповідь вимагає збереження певної последовності фактів або термінів.

На графіку видно, що GPT знову демонструє найвищий результат серед усіх трьох моделей, досягаючи середнього значення ROGUE-L у діапазоні 0.23–0.24. Це свідчить про те, що відповіді, які формує ця модель, не лише містять ключові слова, але й зберігають правильний порядок слів і логіку побудови речення. Така відповідність особливо важлива у складніших завданнях, де просте перерахування фактів недостатнє без правильного контекстного впорядкування.

LLaMA продемонструвала середній результат, де значення ROGUE-L становить близько 0.16. Це нижче за показник GPT, але все ж значно перевищує результат моделі BLOOMZ. Отриманий бал вказує на те, що LLaMA здатна формувати відповіді з правильною структурою, хоча частіше використовує менш точне формулювання або допускає незначні зміни у порядку слів. Ця поведінка може бути пов'язана зі спробами моделі узагальнити або перефразувати контент, що в деяких випадках призводить до втрати прямої відповідності.

BLOOMZ, як і в попередніх метриках, посіла останнє місце. Її показник ROGUE-L дуже низький – на рівні 0.03–0.04. Такий результат свідчить про істотну невідповідність між її відповідями та референсними реченнями не лише за змістом, а й за структурою. Враховуючи, що ця модель часто генерувала короткі або фрагментарні відповіді, низький бал

ROUGE-L є закономірним і підтверджує її обмежену здатність до збереження логічної структури відповіді.

Порівняльний аналіз за ROUGE-L демонструє, що GPT має суттєву перевагу в синтаксичній відповідності. Це дозволяє цій моделі краще імітувати мовні конструкції людини та створювати змістовні і логічні речення. LLaMA, хоч і поступається, все ж демонструє стабільні результати, що дає змогу її розглядати як придатну до задач з високими вимогами до структури. Натомість BLOOMZ не забезпечує належної якості побудови відповідей у контексті цієї метрики.

Загалом, ROUGE-L є показником не просто наявності правильних елементів у відповіді, а й вміння моделі зберігати їх у правильному порядку. В умовах задачі питання–відповідь це відіграє ключову роль, оскільки порушення порядку фактів або граматичної структури може призвести до втрати сенсу. Результати показують, що лише GPT здатна систематично відповідати цим вимогам.

У підсумку можна зробити висновок, що ROUGE-L є важливою доповнюючою метрикою, яка допомагає виявити відмінності у стилі генерації та структурній точності. Її результати корелюють із іншими оцінками, але також відкривають додаткові аспекти якості, які не охоплюються лексичними метриками, такими як BLEU чи ROUGE-1.

У результаті оцінювання відповідей за обраними метриками – BLEU, ROUGE-1, ROUGE-L та BERTScore – було виявлено суттєві відмінності у здатності кожної з моделей до формування релевантних, повноцінних і структурно правильних відповідей. Кожна з моделей продемонструвала свій характерний стиль генерації тексту, що позначилося на підсумкових оцінках.

Найкращі результати за всіма чотирма метриками продемонструвала модель GPT. Вона отримала найвищі середні значення BLEU, ROUGE-1, ROUGE-L і BERTScore, що свідчить про її високу точність, змістову відповідність та граматичну узгодженість. Відповіді GPT були повними,

релевантними, гнучко адаптованими до формату запиту і структурно близькими до еталонних відповідей. Це вказує на загальну перевагу цієї моделі у завданнях типу «питання–відповідь», де критичною є не лише коректність, але й глибина та ясність формулювання.

Модель LLaMA посіла друге місце за більшістю метрик, з помірно високими значеннями ROUGE та BERTScore. Хоча її BLEU-оцінка була нижчою за GPT, вона все ж зберігала певний рівень змістової відповідності та синтаксичної логіки. Однак, через більш обмежену точність формулювання та часткові розбіжності з еталонними відповідями, її загальна якість була дещо нижчою. Тим не менш, модель продемонструвала здатність генерувати логічно зв'язані та змістовно наближені до істини відповіді, що робить її конкурентоспроможною у деяких сценаріях.

Найгірші результати були у моделі BLOOMZ. Її значення BLEU, ROUGE-1 та ROUGE-L виявилися майже нульовими або дуже низькими, що свідчить про суттєву відмінність її відповідей від еталонів як за лексикою, так і за структурою. Хоча BERTScore вказував на певну семантичну схожість, це не компенсувало загальної слабкої відповідності. Часті випадки коротких, неповних або поверхневих відповідей суттєво знижували якість і пояснюють низькі оцінки.

Таким чином, проведений аналіз підтвердив, що GPT є найбільш якісною моделлю для задач питання–відповідь серед розглянутих, тоді як LLaMA показала гідний, але менш точний результат. BLOOMZ виявилась найменш ефективною у контексті оцінювання за класичними метриками, що свідчить про її обмежену здатність до генерації відповідей, релевантних до запиту та контексту. Вибір моделі для подібних завдань має враховувати ці показники, особливо в системах, де точність відповіді має критичне значення.

ВИСНОВКИ

У процесі виконання дослідження на тему «Дослідження великих мовних моделей для завдань із відповідями на питання» було проведено комплексне дослідження, що охоплює теоретичний аналіз, вибір інструментів, експериментальну реалізацію та оцінювання якості роботи сучасних великих мовних моделей (ВММ) у задачах типу «питання–відповідь».

На першому етапі було здійснено ґрунтовний аналіз предметної області, у ході якого досліджено особливості архітектур ВММ, принципи їхнього навчання, основні підходи до реалізації задач відповідей на питання, а також розглянуто переваги й обмеження такого типу систем. Визначено, що задача генерації відповіді вимагає як якісного розуміння вхідного контексту, так і здатності сформувати синтаксично та семантично узгоджену відповідь.

Було обґрунтовано актуальність теми у світлі зростаючого попиту на автоматизовані системи обробки природної мови та пошук ефективних інструментів для їх створення. Особливу увагу приділено нормативним аспектам використання ВММ, включно із законодавчими актами України у сфері захисту персональних даних, авторських прав та електронної взаємодії.

У рамках дослідження було обрано три представницькі моделі для порівняння: BLOOM, LLaMA та GPT. Для кожної з моделей було описано її технічні особливості, архітектурну основу, переваги та потенційні обмеження. В якості тестового набору даних обрано відкритий датасет Natural Questions від Google, який містить реальні запитання та відповідні фрагменти з текстових джерел.

Проведено формалізацію структури експерименту, включно з описом формату вхідних даних, умов виклику моделей, організації збереження результатів, а також обсягом вибірки. Для кожної моделі було реалізовано

окремий скрипт, який автоматично здійснював парсинг даних, генерацію промптів, обробку відповідей та збереження результатів у стандартизованому форматі.

На практичному етапі дослідження проведено запуск моделей у локальному середовищі або через API-платформи, з урахуванням вимог кожної моделі. Було враховано такі фактори, як необхідність попередньої авторизації (LLaMA), платний доступ (GPT), а також апаратні вимоги до локального запуску (особливо для LLaMA). Особлива увага приділялася проблемам продуктивності, часу обробки запитів та технічним обмеженням.

Наступним етапом стало проведення автоматичного оцінювання згенерованих відповідей за допомогою сучасних метрик: BLEU, ROUGE-1, ROUGE-L та BERTScore. Було створено скрипти для обробки відповідей та побудови візуалізацій, що дозволило отримати об'єктивну порівняльну характеристику якості роботи кожної моделі.

У результаті дослідження встановлено, що модель GPT продемонструвала найвищі показники за всіма метриками, забезпечуючи точні, повні та релевантні відповіді. Модель LLaMA показала гідний рівень якості, проте зі значно вищими витратами ресурсів і часу. Модель BLOOM мала суттєво нижчі результати, що свідчить про її обмежену здатність до генерації якісних відповідей у даному завданні.

Таким чином, у ході дослідження було реалізовано повний цикл аналізу ВММ у задачі питання–відповідь: від теоретичного обґрунтування і вибору моделей, до запуску, автоматичного оцінювання та інтерпретації результатів. Отримані результати можуть бути використані для подальшого вдосконалення систем генерації відповідей, оптимізації їх архітектур та вибору моделей залежно від практичних вимог до якості, ресурсів і часу виконання.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Kafedra shtuchnoho intelektu KHNURE. *NURE*. URL: <https://nure.ua/department/kafedra-shtuchnogo-intelektu> (date of access: 04.04.2025).
2. Artificial intelligence department. *NURE*. URL: <https://ai.nure.ua/> (date of access: 04.04.2025).
3. Pro avtorske pravo i sumizhni prava. *Ofitsiyni webportal parlamentu Ukrainy*. URL: <https://zakon.rada.gov.ua/laws/show/2811-20#Text> (date of access: 04.04.2025).
4. Pro zakhyst personal'nykh danykh. *Ofitsiyni webportal parlamentu Ukrainy*. URL: <https://zakon.rada.gov.ua/laws/show/2297-17> (date of access: 04.06.2025).
5. Pro zakhyst prav spozhyvachiv. *Ofitsiyni webportal parlamentu Ukrainy*. URL: <https://zakon.rada.gov.ua/laws/show/1023-12> (date of access: 04.04.2025).
6. Pro elektronnu identyfikatsiiu ta elektronni dovirchi posluhy. *Ofitsiyni webportal parlamentu Ukrainy*. URL: <https://zakon.rada.gov.ua/laws/show/2155-19> (date of access: 04.04.2025).
7. OpenAI. GPT-4 Technical Report. *arXiv*. 2023. Abs/2303.08774. P. 100. URL: <https://arxiv.org/abs/2303.08774> (date of access: 04.04.2025).
8. Industry Leading, Open-Source AI | Llama by Meta. *Meta*. URL: <https://ai.meta.com/llama/> (date of access: 04.04.2025).
9. Bigscience/bloom Model Card. Hugging Face. *Hugging Face*. URL: <https://huggingface.co/bigscience/bloom> (date of access: 04.04.2025).
10. BERTScore: Evaluating Text Generation with BERT / T. Zhang et al. *arXiv*. 2019. Vol. 1904.09675. P. 43. URL: <https://arxiv.org/abs/1904.09675> (date of access: 04.04.2025).
11. Pro skhvalennia Kontseptsii rozvytku shtuchnoho intelektu v Ukraini. *Ofitsiyni webportal parlamentu Ukrainy*.

URL: <https://zakon.rada.gov.ua/laws/show/1556-2020-p#n8> (date of access: 04.04.2025).