

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА Пояснювальна записка

рівень вищої освіти другий (магістерський)

**РОЗРОБКА І ДОСЛІДЖЕННЯ МЕТОДУ «ДОНАВЧАННЯ» МОВНОЇ
МОДЕЛІ ГРТ НА ВЛАСНОМУ КОНТЕНТІ ДЛЯ СТВОРЕННЯ
НЕЙРОКОНСУЛЬТАНТА ДЛЯ АБИТУРІЄНТІВ УНІВЕРСИТЕТУ**

(тема)

Виконав:
студент 2 курсу, групи ІНФМ-22-1

Кіріченко О.Д.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Яковлева О.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____
(підпис)

Кобилін О.А.
(прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)Кафедра Інформатики
(повна назва)Рівень вищої освіти другий (магістерський)Спеціальність 122 Комп'ютерні науки
(код і повна назва)Тип програми освітньо-професійнаОсвітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«_____» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУстудентові Кіріченку Олександр Дмитровичу
(прізвище, ім'я, по батькові)1. Тема роботи розробка і дослідження методу «донавчання» мовної моделі GPT на власному контенті для створення нейроконсультанта для абітурієнтів університету

затверджена наказом по університету від 3 листопада 2023 року № 1280См

2. Термін подання студентом роботи до екзаменаційної комісії 01 січня 2023 р.3. Вихідні дані до роботи науково-методична література, інформація для створення датасету та промптів, language models, архітектура Transformer, LLM GPT, перелік використаних програмних засобів: OpenAPI, Langchain, мова програмування Python, середовище розробки Google Colab.

4. Перелік питань, що потрібно опрацювати в роботі _____

1. Актуальність розробки сервісів на основі мовних моделей.

2. Розробка та дослідження методу навчання моделі gpt3.5 на власному контенті.

3. Розробка нейроконсультанта для абітурієнтів спеціальності комп'ютерні науки, освітньо-професійна програма інформатика.

4. Аналіз та опрацювання результатів нейроконсультанта.

5. Програмна реалізація нейроконсультанта дослідження та створення логіки її роботи.

6. Тестування розробленого нейроконсультанта та аналіз результатів проведеного дослідження.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність сфери розробки нейроконсультанта та його роль в сучасному часу, ілюстрація роботи нейроконсультанта, аналіз роботи, складання датасету, промптів, ілюстрація роботи розробленого нейроконсультанта.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	03.11.2023	
2	Аналіз завдання, підбір літератури	03.11.23-6.11.23	
3	Аналіз літератури з досліджуваної проблеми	7.11.23-12.11.23	
4	Аналіз технічних засобів	13.11.23-14.11.23	
5	Розробка методу	14.12.23-16.12.23	
6	Програмна реалізація	14.12.23-19.12.23	
7	Оформлення пояснювальної записки	10.12.23-19.12.23	
8	Перевірка на плагіат	01.12.2023	
9	Рецензування	02.12.2023	
10	Підготовка презентації та доповіді	05.12.2023	
11	Занесення роботи в електронний архів	10.01.2024	
12	Попередній захист кваліфікаційної роботи	10.01.2024	

Дата видачі завдання 3 листопада 2023 р.

Студент _____
(підпис)

Керівник роботи _____
(підпис)

доц. Яковлева О.В.
(посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 82 с., 1 табл., 41 рис., 3 дод., 16 джерел.

OPENAI, CHATGPT, LANGCHAIN, PROMT, LANGUAGE MODEL, DATASET.

Об'єктом дослідження являється розробка і дослідження можливості «донавчання» мовної моделі ChatGPT на власному контенті для вирішення задачі створення нейроконсультанта для абітурієнтів.

Метою дослідження є створення нейроконсультанта для відповідей на запити абітурієнтів щодо інформації вебсайту Харківського Національного Університету Радіоелектроніки.

Був створений датасет (власний контент), на основі якого формувалися відповіді моделі GPT для абітурієнтів університету. У результаті дослідження здійснена програмна реалізація системи для відповідей мовної моделі GPT на основі власного контенту.

OPENAI, CHATGPT, LANGCHAIN, PROMT, LANGUAGE MODEL, DATASET.

The object of the study is the development and study of the possibility of «retraining» the ChatGPT language model on its own content to solve the problem of creating a neuroconsultant for applicants.

The purpose of the study is to create a neuroconsultant for answering applicants' requests regarding information on the Kharkiv National University of Radioelectronics website.

A dataset (own content) was created, on the basis of which the answers of the GPT model for university applicants were formed. As a result of the research, the software implementation of the system for the answers of the GPT language model based on its own content was carried out.

ЗМІСТ

Вступ.....	8
1 Сучасний стан питання розробки сервісів на основі мовних моделей	11
1.1 Успіхи штучного інтелекту в вирішенні задач обробки природної мови та комп'ютерного зору	12
1.2 Особливості мовних моделей (transformer).....	15
1.3 Моделі GPT та чат бот ChatGPT.....	24
1.4 Розробка сервісів на основі API від GPT	27
1.5 Постановка задачі дослідження	32
2 Розробка та дослідження методу навчання моделі GPT на власному контенті	33
2.1 Використання GPT через API для відповідей на основі власного контенту	33
2.2 Особливості використання GPT через API	35
2.3 Використання фреймворку LangChain для визначення фрагментів, що відповідають запиту користувача.....	36
2.4 Параметри налаштування моделі GPT щодо формування відповіді.....	38
2.5 Алгоритм доначання на власному контенті.....	45
3 Розробка нейроконсультанта для абітурієнтів спеціальності Комп'ютерні науки, освітньо-професійна програма Інформатика	47
3.1 Налаштування програмного середовища для досліджень	47
3.2 Підготовка власного контенту на основі інформації з сайту кафедри Інформатики	51
3.3 Реалізація розробленого алгоритму щодо створення нейроконсультанта	51
3.4 Формування промптів та налаштування параметрів моделі GPT та Langchain	59
3.5 Ілюстрація роботи нейроконсультанта	65

	6
3.6 Оцінка роботи нейроконсультанта групою експертів.....	67
Висновки.....	68
Перелік джерел посилання	70
Додаток А. Приклад матеріалу з сайту кафедри інформатики для формування власного контенту	72
Додаток Б. Фрагменти підготовленого контенту для «донавчання» моделі GPT	73
Додаток В. Приклади відповідей моделі GPT 3.5-turbo на основі підготовленого контенту.....	76

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API – Application Programming Interface (інтерфейс прикладного програмування)

DS – Data set (набір даних)

ШІ – Штучний Інтелект

NLP – Natural language Processing (обробка природної мови)

GPT – Generative Pre-trained Transformer (генеративний попередньо навчаний трансформер)

CV – Computer Vision (комп'ютерний зір)

RLHF – Reinforcement Learning from Human Feedback (навчання з підкріпленням від людини)

LLM – Large Language Model (велика мовна модель)

ВСТУП

Дослідження присвячене питанню використання моделей GPT від OpenAI для формування відповідей на основі власного контенту.

Моделі, подібні до ChatGPT, ґрунтуються на принципах трансформерів, які є різновидом архітектури нейронних мереж, спеціально розроблених для обробки послідовних даних, особливо тексту.

Такі моделі мають архітектуру трансформера. Трансформери використовують механізм уваги (attention mechanism), який дозволяє моделі фокусуватися на різних частинах вхідних даних під час виконання завдання. Це особливо корисно в мовних моделях для розуміння контексту і залежностей у тексті. Моделі, подібні до ChatGPT, спочатку навчаються на великих обсягах текстових даних. Цей процес називається попереднім навчанням і дає змогу моделі «розуміти» мову. Після цього модель може бути додатково налаштована (fine-tuned) на більш специфічних даних для конкретних завдань. Моделі, як ChatGPT, оптимізовані для різних завдань обробки природної мови (Natural Language Processing, NLP), таких як генерація тексту, переклад, відповіді на запитання та інші. ChatGPT і подібні моделі здатні генерувати когерентний і релевантний текст, продовжуючи речення, абзаци або навіть створюючи цілі статті на основі початкового запиту. Моделі використовують навчання з підкріпленням від людини (Reinforcement Learning from Human Feedback, RLHF). Цей метод навчання включає в себе використання зворотного зв'язку від людини для подальшого поліпшення продуктивності моделі в конкретних сценаріях використання. Моделі, подібні до ChatGPT, являють собою передові досягнення в галузі штучного інтелекту і машинного навчання, демонструючи високу здатність до виконання безлічі завдань з NLP, таких як:

- генерація тексту (моделі можуть продовжувати речення, параграфи або навіть генерувати цілі статті на основі початкового запиту);

- переклад мови (навчені на текстах різними мовами, моделі GPT можуть перекладати текст з однієї мови на іншу);
- відповіді на запитання (моделі можуть відповідати на запитання, надаючи інформацію або пояснюючи концепції);
- сумаризація текстів (мовні моделі можуть коротко викладати основні моменти великих текстів);
- діалоги (моделі GPT можуть підтримувати діалоги, моделюючи бесіди на задану тему);
- кодування (вони можуть генерувати і виправляти програмний код на основі заданих специфікацій);
- ігри в слова і головоломки (мовні моделі можуть розв’язувати кросворди і грати в словесні ігри) та інше.

Вихід моделей GPT (Generative Pretrained Transformer), особливо ChatGPT від OpenAI, справив значний вплив на світ штучного інтелекту і технологій, привносячи революційні поліпшення в сферу NLP. ChatGPT швидко стала популярною серед користувачів і розробників завдяки своїй універсальності, можливості вести бесіду, надавати інформацію та створювати творчі тексти. Це стимулювало подальші дослідження і розробки в галузі штучного інтелекту та призвело до появи нових етичних і практичних питань, пов’язаних із використанням ШІ. Також виникли обговорення щодо впливу цих технологій на трудовий ринок та освіту, їхньої ролі в автоматизації та зміні природи професій, а також щодо потенційної інтеграції в освітній процес. Загалом, поява ChatGPT та інших моделей GPT ознаменувала нову епоху в розвитку ШІ, надавши потужні інструменти для аналізу та генерації природної мови і відкривши нові горизонти в різних галузях.

В роботі розглянути підходи щодо «донавчання» мовних моделей GPT на власному контенті. Розглянута бібліотека LangChain для пошуку фрагментів датасету, що якнайкраще відповідають запиту користувача. Досліджена проблема формування та налаштування промптів для моделей GPT.

Результати досліджень було використано для розробки нейроконсультанта для абітурієнтів університету. Для чого був створений датасет (власний контент), на основі якого формувалися відповіді моделі GPT для абітурієнтів університету. Експертна оцінка була використана для визначення якості відповідей нейроконсультанта. Результати показали суттєві перспективи використання запропонованого підходу для формування відповідей моделей GPT від OpenAI на основі власного контенту.

Актуальність дослідження полягає в тому, щоб спростити пошук основної інформації для абітурієнтів, більш розширених відповідей. Що в результаті позитивно вплине на бажання абітурієнтів користуватися послугами ХНУРЕ.

1 СУЧАСНИЙ СТАН ПИТАННЯ РОЗРОБКИ СЕРВІСІВ НА ОСНОВІ МОВНИХ МОДЕЛЕЙ

1.1 Успіхи штучного інтелекту в вирішенні задач обробки природної мови та комп'ютерного зору

Штучний інтелект (ШІ) – це надзвичайно важлива і динамічна галузь технологій, яка має великий потенціал для зміни нашого світу. Розвиток ШІ в прозі може бути порівняний з епопеєю науки, технології та інновацій, яка неперервно відкриває перед нами нові горизонти.

Уявімо, що ШІ – це розумний персонаж, який допомагає героям нашого світу робити неймовірні речі. Він є важливим союзником інженерів, дослідників та інноваторів, сприяючи розв'язанню складних проблем і реалізації захоплюючих ідей.

Розвиток ШІ може бути подібним до чарівницького чарівництва, де комп'ютери набувають розуму та здатності навчатися самостійно. Вони стають майстрами обробки природної мови, розпізнавання облич, вирішення завдань машинного навчання та генерації тексту. Це дозволяє їм здійснювати речі, які колись були мрією.

У нашій розповіді розвиток ШІ може бути порівняний з епохою досліджень та відкриттів, коли вчені і інженери долали грані знань і створювали нові технології. Вони вчили машини розуміти нашу мову, аналізувати великі обсяги даних і допомагати людям вирішувати глобальні проблеми, такі як зміна клімату, медичні діагностики та розвиток науки.

Розвиток ШІ може залишити за собою питання етики, приватності і безпеки. Важливо забезпечити, щоб наш розумний ШІ діяв у благородних інтересах і не завдавав шкоди суспільству.

Усі ці аспекти ШІ роблять його незворотнім відкриттям, яке формує майбутнє нашої цивілізації. Розвиток ШІ в прозі може служити джерелом

незрівняної інспірації, але також вимагає обачності та розуміння важливості етики і доброчесності в сфері технологій.

Нижченаведено ключові аспекти розвитку ШІ:

– машинне навчання, машинне навчання є основним підходом до створення ШІ-систем. Воно включає в себе використання алгоритмів, які навчаються на основі даних і можуть самостійно вдосконалювати свою продуктивність з часом. Глибокі нейронні мережі, такі як трансформери, стали ключовим інструментом у цьому процесі;

– обробка природної мови (NLP), розвиток NLP дозволяє системам розуміти і генерувати природну мову. Мовні моделі, такі як GPT, здатні створювати високоякісний текст, розпізнавати інтенцію користувачів та виконувати багато інших завдань NLP;

– комп'ютерний зір (CV), CV дозволяє комп'ютерам розпізнавати об'єкти на зображеннях і відео, а також розуміти глибокі структури інформації в візуальних даних. Це має застосування в автомобільній промисловості, медицині, безпеці та багатьох інших галузях;

– зростання обчислювальної потужності, завдяки розвитку апаратних технологій, таких як графічні процесори (GPU) і спеціалізовані прискорювачі для ШІ, дослідники мають доступ до більших обчислювальних ресурсів, що дозволяє створювати більш потужні моделі і швидше навчати їх;

– застосування в різних галузях. ШІ застосовується в автономних автомобілях, медицині (діагностика та лікування), фінансах (прогнозування ринків та управління ризиками), робототехніці, енергетиці, сфері обслуговування та багатьох інших галузях;

– етичні питання з ростом можливостей ШІ постають важливі етичні питання, такі як приватність даних, вплив на робочі місця та суспільство, а також питання відповідальності за прийняття рішень;

– регулювання і законодавство, уряди і регулятори по всьому світу активно працюють над створенням законів і нормативів, які регулюють використання ШІ, щоб забезпечити безпеку та ефективність цих технологій.

Близко року тому відбувся прорив у розвитку мовних моделей. Це відбулося 22 листопада 2022, коли компанія OpenAI презентувала чат бот ChatGPT, який має здатність взаємодіяти у формі розмовного діалогу і надавати відповіді, які можуть здатися людяними. Він базується на мовної моделі GPT-3.5 turbo, що відноситься до моделей LLM (Large Language Model) і використовує метод навчання з підкріпленням із зворотнім зв'язком від людей (Reinforcement Learning with Human Feedback-RLHF). Це додатковий рівень навчання, в якому співпрацює людина для того, щоб навчити модель дотримуватися вказівок і генерувати відповіді, які здаватимуться більш живими і прийнятими до людей.

GPT-3.5 turbo стала однією з найпотужніших LLM моделей у світі. Вона працює на основі 175 мільярдів параметрів і була навчена, використовуючи близько двох третин загального обсягу даних зі всесвітньої мережі [1].

Технологія генеративного штучного інтелекту дозволяє створювати унікальний контент, використовуючи обширний обсяг даних і методи машинного навчання. На ринку існує багато аналогічних інструментів, і ChatGPT – лише один із них. Наприклад, інші відомі моделі на основі нейромереж, DALL-E та Midjourney, здатні створювати зображення на основі текстових описів. Але всі ці успіхи ґрунтуються на більш ніж 60 річних дослідження щодо класичних підходів, які базуються на інформаційно-аналітичних методах, та більш сучасних нейромережевих підходах в областях Computer Vision та Natural Language Processing, наприклад, [2–9].

В порівнянні з іншими популярними платформами ChatGPT розвивається неймовірно швидко. Він досяг мільйонного користувача всього за п'ять днів, що на 70 днів швидше, ніж Instagram, який досяг 1 мільйона користувачів за 75 днів. Статистику користувачів зображено на рисунку 1.1.

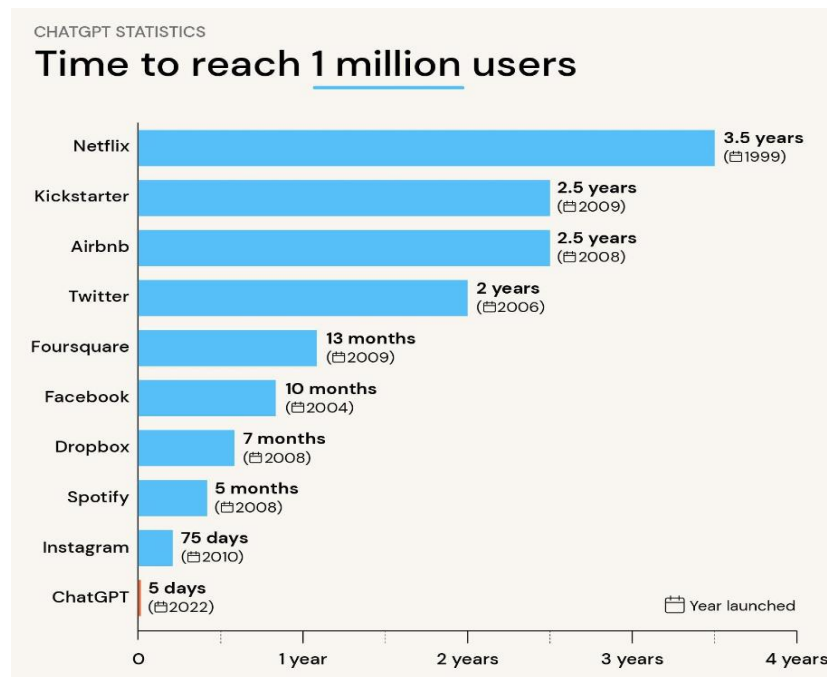


Рисунок 1.1 – Період, за який сервіс ChatGPT досяг одного мільйона користувачів [1]

Проте наразі жодна модель не змогла здобути таку популярність, як модель GPT-3.5 turbo, що була обгорнута у чат-бота, та отримала назву ChatGPT. Така популярність визнана тим, що використання чат-боту ChatGPT є дуже простим, модель здатна давати відповіді як людина та безкоштовна версія доступна для всіх і працює на багатьох мовах. ChatGPT створено на основі моделі GPT-3, яка розроблена ще в 2020 році, але це не перша модель, раніше були ще GPT-1 (Generative Pre-trained Transformer), GPT-2, що з'явилися ще в 2018 та 2019 роках. В березні 2023 року Open AI створила ще більш просунуту модель GPT-4 [10]. На сьогодні ChatGPT працює на основі моделей GPT-3.5 turbo та GPT-4.

Для інтеграції ChatGPT у сторонні сервіси та продукти необхідний API-ключ, який можна отримати в особистому кабінеті на сайті OpenAI [10]. Далі необхідно надіслати моделі запит, що містить ключ API та вхідні дані (питання від користувача та додаткову інструкцію-prompt, як треба моделі формувати відповідь.

Далі в роботі не буде уточнюватися версія моделі, а просто використовуватися назва моделі GPT, тому запропонований підхід, застосовується до обох версій моделей ChatGPT. Як розвивалися моделі ChatGPT показано на рисунку 1.2.

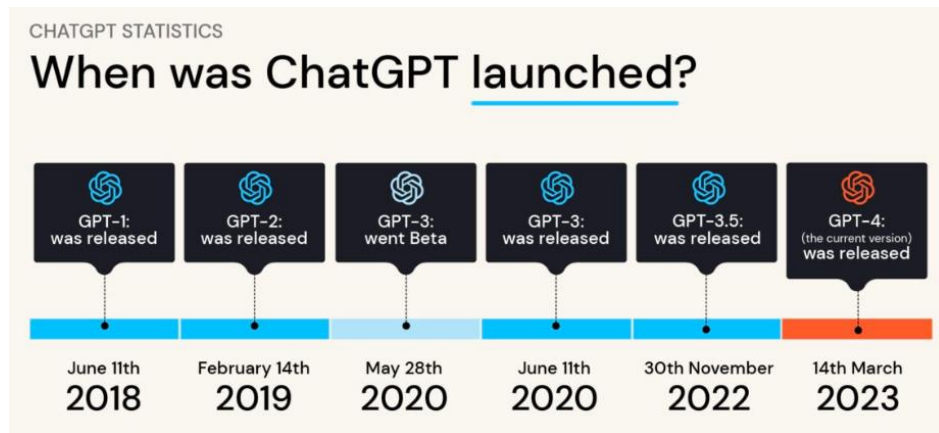


Рисунок 1.2 – Розвиток моделей GPT [1]

1.2 Особливості мовних моделей (transformer)

Мовна модель, зокрема Transformer, – це технологія машинного навчання і обробки природної мови (NLP), яка призначена для розуміння і генерації тексту. Transformer – це одна з найпоширеніших архітектур для роботи з текстовими даними і була запропонована у 2017 році [11].

Основною ідеєю Transformer є використання механізму уваги для ефективного моделювання взаємодії між словами в тексті. Вона дозволяє моделі фокусуватися на різних частинах вхідного тексту, залежно від їх важливості для різних завдань NLP, таких як переклад, генерація тексту, розпізнавання тексту та інші. Архітектуру моделі Transformer зображено на рисунку 1.3.

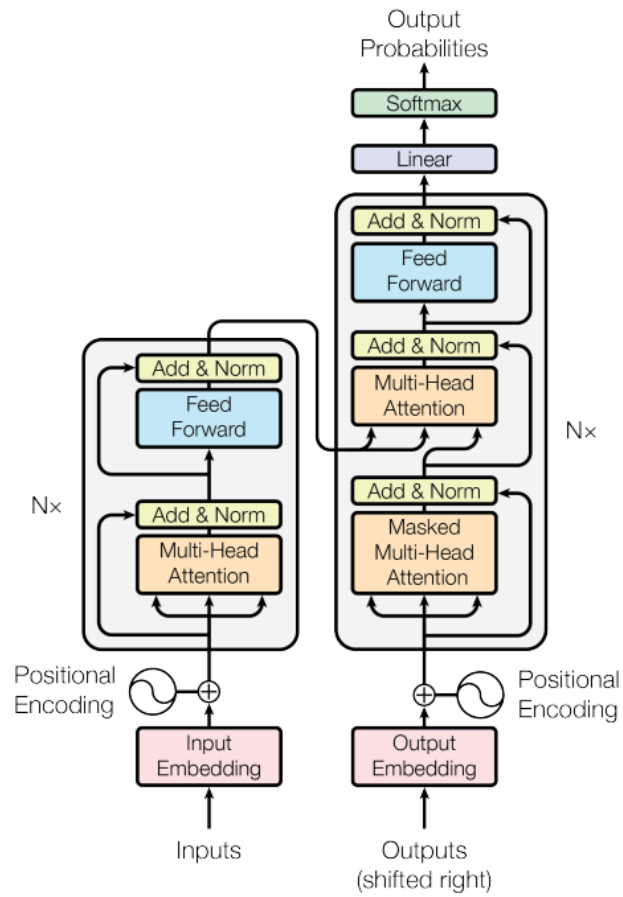


Рисунок 1.3 – Архітектура моделі Transformer [11]

Принцип роботи моделі Transformer на найвищому рівні абстракції зображено на рисунку 1.4.



Рисунок 1.4 – Принцип роботи моделі Transformer на найвищому рівні абстракції [11]

Більш детальніший принцип роботи зображено на рисунку 1.5.

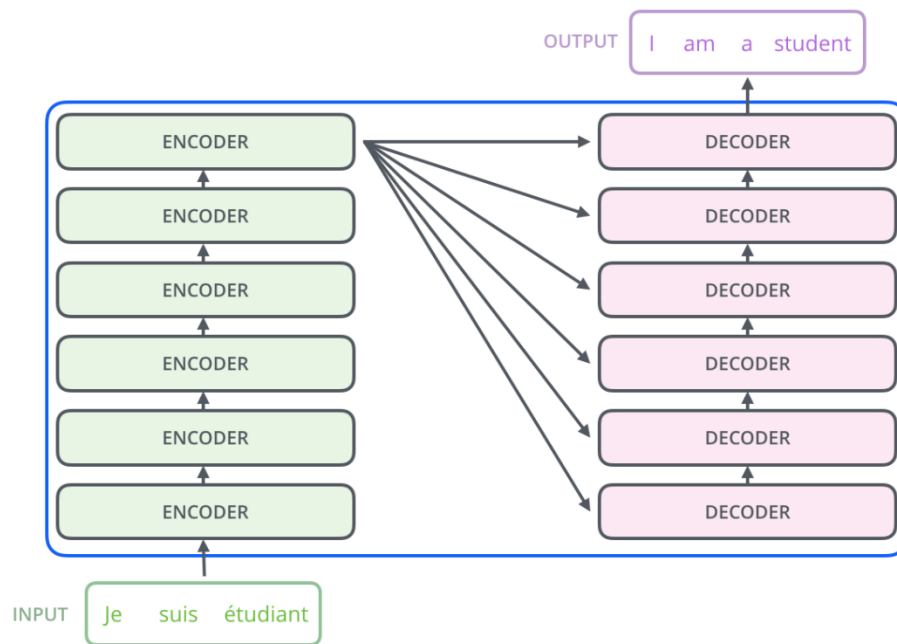


Рисунок 1.5 – Принцип роботи моделі Transformer, дельніше ніж на рис.1.4
найвищому рівні абстракції [11]

Блок кодування являє собою набір енкодерів, на рисунку 1.5 показано 6 енкодерів, розташованих один над іншим (число 6 не є обов'язковим, можна пробувати й інші варіанти). Блок декодування складається з набору декодерів, яких також шість.

Усі енкодери ідентичні за структурою, хоч і мають різну вагу. Кожен можна розділити на два підшари, що зображено на рисунку 1.6.

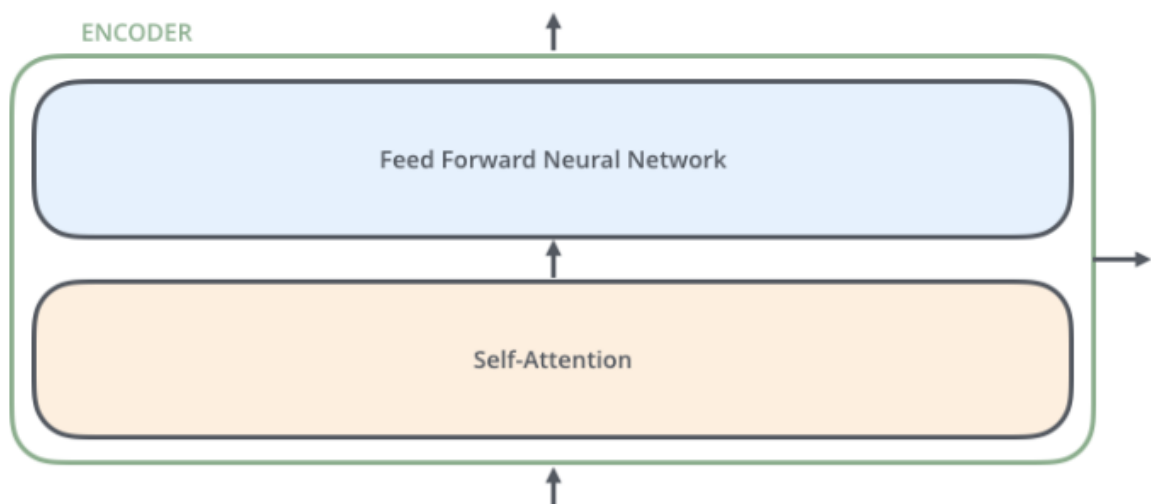


Рисунок 1.6. – Підшари енкодерів [11]

Вхідна послідовність, що надходить в енкодер, спочатку проходить через шар внутрішньої уваги (self-attention), що допомагає енкодеру подивитися на інші слова у вхідному реченні під час кодування конкретного слова. Вихід шару внутрішньої уваги вирушає у нейронну мережу прямого поширення (feed-forward neural network). Така сама мережа незалежно застосовується для кожного слова в реченні.

Декодер також містить два шари, але між ними є шар уваги, який допомагає декодеру фокусуватися на релевантних частинах вхідної пропозиції, що вказано на рисунку 1.7.

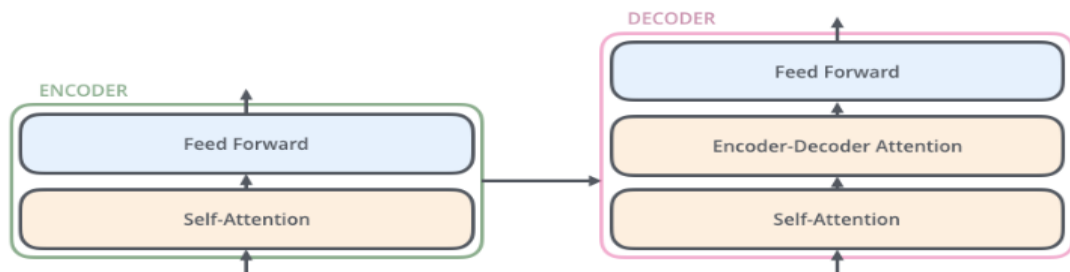


Рисунок 1.7 – Принцип роботи моделі Transformer, найвищому рівні абстракції [11]

Тепер, коли ми побачили основні компоненти моделі, подивимося різні вектори/тензори, і як вони передаються від компонента до компоненту, перетворюючи вхідну послідовність навченої моделі у вихідну.

Як і у разі будь-якого NLP-програми, ми починаємо з того, що перетворюємо слово на вектор, використовуючи алгоритм ембеддингів слів (word embeddings), що вказано на рисунку 1.8.



Рисунок 1.8 – Перетворення слів на вектори використовуючи алгоритм ембеддингів [11]

Ембедінги застосовуються тільки в нижньому енкодері. На рівні абстракції, загальної для всіх енкодерів, відбувається наступне: енкодери отримують набір векторів розмірністю 512 (для найнижчого енкодера це будуть ембедінг слів, для інших - вихідні вектори нижчестоящих енкодерів). Розмір цього набору векторів є гіперпараметром, який ми можемо встановлювати, і, по суті, дорівнює довжині найдовшого речення у навчальному корпусі.

Після того як слова вхідної речення перетворилися на ембедінги, кожен з них окремо проходить через два шари енкодера, що можна побачити на рисунку 1.9.

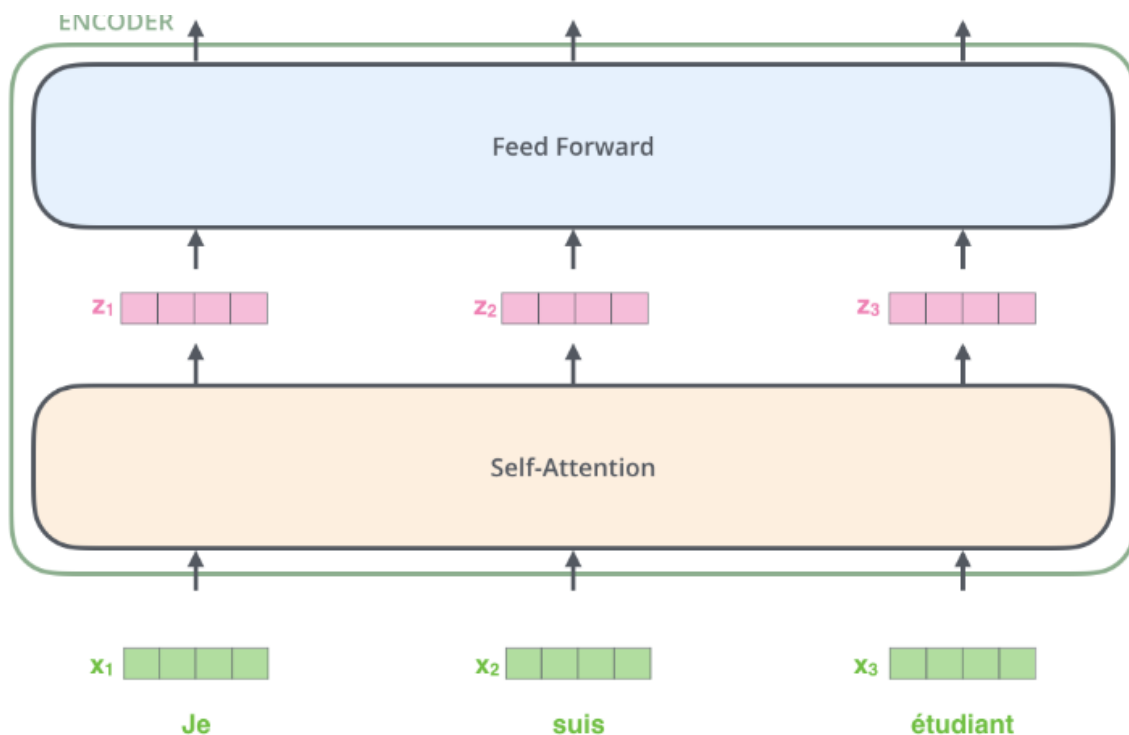


Рисунок 1.9 – Перетворення слів на ембедінги [11]

Тут зображено одну з основних особливостей Трансформера: кожне слово йде своєю власною траєкторією в енкодері. І хоча існують залежності між цими траєкторіями в шарі внутрішньої уваги, в шарі мережі прямого поширення таких залежностей немає, що дозволяє різним траєкторіям виконуватися паралельно під час проходження через цей шар.

Як ми вже згадували, енкодер отримує на вхід і обробляє набір векторів, проводячи їх через шар внутрішньої уваги і далі через нейронну пряму мережу поширення, поки, нарешті, не передає свій вихід наступному енкодеру, що вказано на рисунку 1.10.

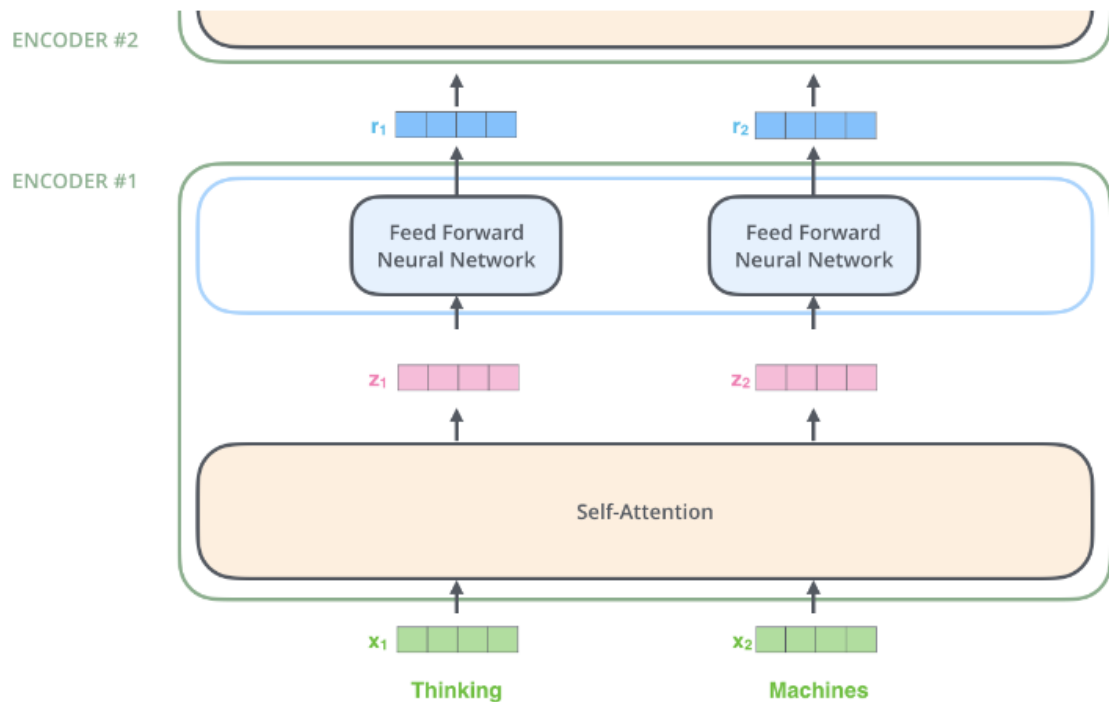


Рисунок 1.10 – Перетворення слів на ембедінги №2 [11]

Таким чином, Transformer складається з двох основних компонентів:

- кодер приймає на вхід послідовність слів або символів і перетворює їх в складний вектор, який захоплює семантику тексту та його структуру;
- декодер використовує для генерації вихідного тексту або відповіді на певне завдання, наприклад, машинний переклад або генерацію тексту.

Основні компоненти моделі Transformer включають в себе:

- механізм вимірювань уваги (Attention Mechanism)– це ключова частина, яка дозволяє моделі фокусуватися на різних частинах вхідного тексту залежно від їхнього контексту і важливості;
- позиційна ембедінг (Positional Embedding), для того, щоб модель могла розрізнити слова в послідовності, до входу додаються вектори, які вказують на

позицію слова в тексті;

- множина вихідних шарів (Multiple Encoder and Decoder Layers), Transformer складається з кількох повторюваних шарів, які обробляють вхідний текст послідовності за послідовним;

- маски (Masks) використовуються для приховування деяких частин вхідного тексту під час навчання, що допомагає моделі розрізняти контекст;

- feedforward мережі, кожен шар має фінальний шар з функцією активації, яка допомагає виконувати обчислення.

Моделі Transformer були успішно використані в багатьох NLP-завданнях, таких як машинний переклад, розпізнавання мови, генерація тексту, аналіз тональності текстів і багато інших. Вони стали основою для створення величезних мовних моделей, таких як GPT (Generative Pre-trained Transformer) від OpenAI і BERT (Bidirectional Encoder Representations from Transformers) від Google, які показали вражаючі результати в багатьох NLP-завданнях завдяки своїм здатностям до розуміння і генерації природної мови.

Особливостями мовних моделей вважаються:

- паралельна обробка послідовностей, Transformer здатний обробляти послідовності паралельно, що робить його набагато швидшим у порівнянні з традиційними рекурентними мережами, де обчислення проводяться послідовно. Це робить Transformer дуже підходящим для обробки великих обсягів текстових даних;

- архітектура без рекурентних зв'язків. У відміну від багатьох рекурентних нейронних мереж, у Transformer немає рекурентних зв'язків. Це дозволяє йому обробляти текст паралельно та ефективно і позбавляє його від проблем з градієнтами, які можуть виникнути у рекурентних мережах під час тренування;

- масштабованість Transformer легко масштабується до великих розмірів даних і задач, завдяки можливості додавання більшої кількості шарів і голів або збільшення розмірів векторів у вбудовах;

- претреновані моделі Transformer дозволив створити потужні

претреновані мовні моделі, такі як GPT (Generative Pre-trained Transformer) і BERT (Bidirectional Encoder Representations from Transformers). Ці моделі навчаються на великих обсягах текстових даних і можуть бути подальше доналаштовані для конкретних завдань, що робить їх дуже ефективними в різних NLP задачах.

Мовні моделі грають ключову роль у розвитку багатьох сучасних NLP-застосунків і продуктів, і вони постійно покращуються завдяки новим методам і даним. Наразі ми можемо відмитити таку важливу актуальність як актуальність розробки сервісів на основі мовних моделей. Вона лишається дуже високою і продовжує зростати. Ось декілька ключових факторів, які вказують на цю актуальність:

- зростання обробки природної мови (NLP). Застосунки обробки природної мови стають все більш розповсюдженими в бізнесі та технологіях. Мовні моделі є основою для багатьох NLP-завдань, таких як автоматичний переклад, аналіз настроїв, розпізнавання мови та інші;

- зростання якості моделей. Завдяки великим обсягам даних та покращеним архітектурам, мовні моделі стають все точнішими і здатними вирішувати більше завдань. Це робить їх важливими для розробки нових застосунків та сервісів;

- застосування в автоматизації, мовні моделі допомагають у вирішенні завдань автоматизації, таких як автоматичне відповідання на питання, робота з клієнтськими запитами, створення чат-ботів і віртуальних асистентів;

- покращення користувацького досвіду, сервіси на основі мовних моделей можуть покращити користувацький досвід, дозволяючи взаємодіяти з системою за допомогою природної мови, що зробить комунікацію більш інтуїтивною і зручною;

- аналіз даних, мовні моделі також використовуються для аналізу великих обсягів текстових даних, які зберігаються в інтернеті. Вони допомагають в розумінні трендів, виявленні важливої інформації та прийнятті бізнес-рішень на основі цих даних.

Загалом, мовні моделі є ключовим елементом розвитку інтелектуальних систем, і їхні можливості стають все ширше використовуваними в різних галузях індустрії.

Розробка сервісів на основі мовних моделей залишається дуже актуальною і важливою галуззю в області інформаційних технологій і обробки природної мови. Ось деякі ключові аспекти, що підкреслюють актуальність цих розробок:

- машинний переклад, сервіси машинного перекладу на основі мовних моделей продовжують поліпшуватися і стають доступними для більшої кількості мов. Це допомагає сприяти комунікації між різними культурами та розширювати глобальні можливості бізнесу і спілкування;

- голосовий асистент, мовні моделі використовуються для реалізації голосових асистентів, які надають інтерактивну допомогу користувачам у різних аспектах їхнього життя, включаючи навігацію, пошук інформації та виконання завдань;

- автоматичне завершення тексту, мовні моделі допомагають під час написання текстів, пропонуючи автоматичне завершення фраз і прогнозуючи наступне слово, що полегшує процес створення контенту;

- аналіз настрою і сентименту, мовні моделі дозволяють аналізувати та розуміти сентимент та настрій текстів, що є важливим для брендів та компаній, які відслідковують публічну думку про свої продукти або послуги;

- генерація контенту, моделі генерації тексту, такі як GPT (Generative Pre-trained Transformer), можуть створювати структурований і якісний контент автоматично, що корисно для блогерів, журналістів та багатьох інших.

- ефективний пошук інформації, мовні моделі використовуються для покращення пошукових систем, забезпечуючи більш точні та релевантні результати пошуку;

- освіта, використання мовних моделей може полегшити процес навчання та розвитку навчальних матеріалів, включаючи індивідуальний підхід до навчання;

– клієнтське обслуговування при використанні чат-ботів на основі мовних моделей може полегшити обробку запитів клієнтів та підтримку користувачів;

– боротьба з дезінформацією і фейками, моделі можуть допомогти в розпізнаванні фейків та маніпуляційній інформації в мережі, що стає все більш важливим завданням;

За рахунок постійного вдосконалення технологій і навчання на великих обсягах даних, мовні моделі стають більш потужними і корисними для багатьох сфер життя і бізнесу.

1.3 Моделі GPT та чат бот ChatGPT

Моделі GPT (Generative Pre-trained Transformer) та чат-бот ChatGPT базуються на технології Transformer і розроблені OpenAI. Це генеративні мовні моделі, які навчаються передбачати наступні слова або токени в послідовності тексту на основі контексту попередніх слів.

Вони базуються на архітектурі Transformer, що дозволяє їм моделювати взаємозв'язки між словами в тексті і створювати послідовності тексту з вражаючими мовними навичками.

GPT-3, зокрема, є найбільшою і потужною моделлю з серії GPT, маючи майже 175 мільярдів параметрів. Вона здатна виконувати різноманітні завдання, такі як генерація тексту, машинний переклад, відповіді на запитання, розпізнавання мови і багато інших.

ChatGPT – це конкретна реалізація GPT для створення чат-ботів і систем інтерактивного спілкування з користувачами у формі текстового діалогу.

Він заснований на моделі GPT-3, яка навчилася відповідати на запитання та спілкуватися з людьми у текстовому форматі. ChatGPT може використовуватися для створення чат-ботів, віртуальних асистентів, та інших текстових інтерфейсів.

OpenAI пропонує платну послугу «ChatGPT Plus,» яка надає підписку на користування ChatGPT і додаткові переваги для користувачів.

Історія моделей GPT і ChatGPT – це захоплюючий шлях розвитку штучного інтелекту і обробки природної мови.

Поява Transformers, важливим етапом у розвитку моделей GPT була поява архітектури Transformers в 2017 році. Ця архітектура вперше була розроблена у Google і вражаюче покращила здатність моделей до обробки послідовностей, таких як текст.

Перша ітерація моделі GPT, відома як GPT-1, була представлена в жовтні 2018 року OpenAI. Ця модель мала 117 мільйонів параметрів і вже тоді вражала здатністю генерувати текст та виконувати завдання обробки природної мови.

У лютому 2019 року OpenAI оголосила про модель GPT-2, яка вражала своєю здатністю генерувати дуже реалістичний текст. За страхом можливого зловживання, OpenAI спочатку не випустила модель повністю, але пізніше вона стала доступною громадськості. GPT-2 відрізняється від GPT-1 більшою кількістю параметрів і здатністю генерувати більш довгий інформативний текст.

У червні 2020 року OpenAI представила GPT-3, яка стала найпотужнішою моделлю GPT на той момент з майже 175 мільярдами параметрів. GPT-3 вражає здатністю виконувати різноманітні завдання обробки природної мови, включаючи генерацію тексту, машинний переклад, розпізнавання мови та багато інших.

Проєкт «ChatGPT» в OpenAI: у березні 2019 року OpenAI почала розробку проєкту «ChatGPT» як частини більшої програми «AI in Education» спрямованої на використання штучного інтелекту для покращення освіти.

Запуск ChatGPT у вересні 2019: у вересні 2019 року був випущений перший прототип ChatGPT, який був доступний для публічного тестування. Він дозволяв користувачам спілкуватися з моделлю у формі текстового чату.

У листопаді 2020 року OpenAI оголосила про запуск ChatGPT,

побудованого на базі GPT-3. ChatGPT став доступним для використання після введення програми «ChatGPT Plus», яка надає платні підписки на спілкування з ботом та інші переваги.

ChatGPT став популярним інструментом для створення чат-ботів, спілкування зі штучними асистентами та інших текстових засобів комунікації.

Чат-боти на основі моделі GPT, такі як ChatGPT, можуть бути використані для багатьох різних цілей у сфері обробки природної мови. Вони дозволяють підтримувати текстовий діалог між користувачами та комп'ютером або системою, не вимагаючи наявності живої людини на іншому кінці лінії. Додатково до вищесказаного ми можемо додати наступні функції чату:

- підтримка клієнтів. Багато компаній використовують чат-ботів для надання швидкої інформації та відповідей на запитання клієнтів. Це полегшує доступ до служби підтримки та допомагає вирішувати звичайні запити;

- інтерактивний вебсайт. Чат-боти можуть бути вбудовані на вебсайти, щоб надавати відвідувачам інтерактивну можливість звертатися до них з запитаннями, дізнаватися інформацію або навіть керувати процесами на вебсайті;

- розважальні додатки. Чат-боти можуть створювати розважальні інтеракції для користувачів, такі як ігри, історії або жарти, роблячи користування додатком більш цікавим і веселим;

- доступ до інформації. Чат-боти можуть шукати та надавати інформацію з Інтернету, включаючи відповіді на запитання, статті, новини та багато іншого;

- тестування продуктів і ідей. Вони можуть використовуватися для проведення опитувань користувачів, отримання відгуків та тестування нових ідей чи продуктів;

- персональні асистенти. Чат-боти можуть допомагати користувачам у щоденних справах, нагадувати про події, створювати списки завдань та багато іншого.

Отже, чат-боти на базі GPT є потужними інструментами для полегшення спілкування між людьми та комп'ютерами у формі текстового діалогу і мають безліч застосувань у бізнесі, освіті, розвагах та інших сферах.

1.4 Розробка сервісів на основі API від GPT

API (Application Programming Interface) – це набір правил і протоколів, що дозволяють програмам взаємодіяти одна з одною. API надає інтерфейс для використання функцій або даних інших програм без необхідності знати їхню внутрішню логіку. Важливі характеристики API включають структурованість, автентифікацію, формати даних, документацію та масштабованість. Існують різні типи API, такі як RESTful API, SOAP API, GraphQL API та WebSocket API, кожен з яких має свої особливості та застосування. API використовується в різних сферах розробки програмного забезпечення для забезпечення взаємодії між різними програмами та системами.

API або інтерфейс прикладного програмування, є набором певних інструкцій та стандартів, які дозволяють різним додаткам та системам взаємодіяти один з одним. Це дозволяє розробникам створювати нові програми та послуги, використовуючи вже готові функції, надані іншими програмами та сервісами.

Основними принципами роботи API є простота та надійність. API має бути легко зрозумілим та простим у використанні, щоб розробники могли швидко та без проблем інтегрувати його у свої програми. Також API має бути надійним і стабільним, щоб програми, які його використовують, могли працювати без збоїв і помилок.

Існує кілька видів API, які використовуються в різних застосунках та сервісах. Наприклад, RESTful API (Representational State Transfer) є одним із найпопулярніших типів API. Він використовує стандартні HTTP-запити, такі як GET, POST, PUT та DELETE, для взаємодії з застосунками та сервісами. Ще

одним прикладом API є SOAP (Simple Object Access Protocol). Використання API є невід'ємною частиною сучасної розробки та дозволяє розробникам створювати більш потужні, гнучкі та зручні програми. У цьому дослідженні ми розглянули основні концепції та принципи роботи з API, приклади використання в різних галузях, а також найкращі практики та поради щодо роботи.

Однією з головних причин важливості використання API є те, що він дозволяє інтегрувати різні сервіси та програми, створюючи більш функціональні та зручні продукти для користувачів. Крім того, використання API дозволяє прискорити процес розробки та скоротити витрати на розробку та тестування застосунків.

Розглянемо основні кроки, які необхідно виконати при роботі з API:

– крок 1 – це реєстрація та отримання ключа API. Спочатку, коли ми працюємо з API, нам потрібно зареєструватися на відповідному сервісі та отримати унікальний ключ API. Цей ключ API є спеціальним ідентифікатором, який надає доступ до взаємодії з API. Зазвичай цей ключ виділяється після успішної реєстрації на вебсайті, який надає API;

– крок 2 – це є авторизація для взаємодії з API. Потрібно пройти авторизацію. Зазвичай це виконується за допомогою ключа API, який передається в запитах. Деякі API можуть також вимагати додаткової авторизації, наприклад, за допомогою імені користувача та пароля;

– крок 3 – це робота з документацією API. Документація API містить опис всіх доступних методів і параметрів, які можна використовувати при взаємодії з API. У документації вказуються типи запитів, формати передачі даних та можливі помилки. Перед початком роботи з API дуже важливо докладно ознайомитися з документацією API. Читання документації API є першим етапом перед початком роботи з ним. Вам потрібно вивчити всю доступну інформацію про API. Потім, щоб зрозуміти, як воно працює, ви маєте спробувати надіслати запити та отримати відповіді;

– крок 4 – це тестування запитів та відлагодження. Після того, як ви ознайомилися з документацією, вам потрібно випробувати запити до API та обробку відповідей. Щоб переконатися, що запити правильно формуються, ви можете використовувати інструменти для тестування API, такі як Postman або Swagger. Вони дозволяють надсилати запити до API та отримувати відповіді;

– крок 5 – це є обробка відповідей API. Після отримання відповідей від API вам потрібно обробити ці дані. Для цього необхідно провести парсинг відповідей та витягти з них потрібні дані. Також важливо враховувати можливі помилки, які можуть виникнути при роботі з API, та передбачити обробку цих помилок у вашому програмному коді.

Обробка відповідей API – важливий етап, що дає змогу отримати необхідні дані з відповідей та використовувати їх у вашій програмі. при цьому необхідно враховувати можливі помилки та передбачати їх обробку, щоб уникнути проблем під час роботи з API.

При роботі з API важливо дотримуватися кількох кращих практик, щоб забезпечити ефективність, безпеку та надійність вашого програмного продукту.

Розглянемо найкращі практики при роботі з API:

– організація коду, необхідно структурувати свій код так, щоб він був легко зчитуваним та зрозумілим. Це допоможе спростити взаємодію з API та запобігти помилкам під час розробки;

– оптимізація запитів, необхідно використовувати лише ті дані, які дійсно потрібні для нашого застосунку. Рекомендується також оптимізувати дані для підвищення продуктивності нашої програми;

– кешування, використовуємо кешування для зменшення кількості запитів до API та підвищення продуктивності програмного продукту.

Як уникнути витрат на використання API:

– вибір відповідного тарифного плану. Перед початком роботи з API необхідно оцінити обсяги запитів та вибрати відповідний тарифний план. Деякі API можуть надавати безкоштовний тариф з обмеженою

функціональністю;

– використання лише необхідних запитів: необхідно використовувати лише ті дані, які дійсно потрібні, і уникаючи зайвих запитів для зменшення витрат на використання API.

Можна використовувати такі інструменти для роботи з API:

– Postman – це популярний інструмент для тестування та документування API. Він дозволяє легко надсилати запити та отримувати відповіді, а також створювати документацію;

– Insomnia – це інший популярний інструмент для тестування та налагодження. Він надає широкий набір функцій для роботи з API, включаючи підтримку GraphQL та автоматичну генерацію коду;

– Swagger використовується для розробки та документування API. Він дозволяє створювати специфікації у форматі OpenAPI та автоматично генерувати документацію та код для клієнтських бібліотек.

Використання API є невід’ємною частиною сучасної розробки та дозволяє розробникам створювати більш потужні, гнучкі та зручні програми. Ми розглянули основні концепції та принципи роботи з API, приклади використання в різних галузях, а також найкращі практики та поради щодо роботи.

Однією з головних переваг використання API є можливість інтеграції різних сервісів та програм для створення більш потужних та корисних продуктів для користувачів. Крім того, API сприяє прискоренню розробки та зменшенню витрат на створення та тестування програмних застосунків.

Розробка сервісів на основі API від GPT може бути цікавою та корисною задачею. API GPT (Generative Pre-trained Transformer) дозволяє використовувати розуміння мови та генерацію тексту на високому рівні завдяки розробці моделей, навчених на великих обсягах текстових даних.

Також ми можемо використовувати API для наступних завдань:

– генерація контенту, ми можемо розробити сервіс, який автоматично генерує текст для різних потреб. Наприклад, це може бути генерація статей

для вебсайтів, створення описів товарів для електронного магазину або автоматичне створення текстів для соціальних мереж;

- відповіді на запити, створення вебсервісу, який відповідає на запити користувачів за допомогою зрозумілої та контекстної мови. Наприклад, ми можемо розробити віртуального помічника, який відповідає на питання користувачів або створює автоматичні відповіді на листи електронної пошти;

- переклад тексту. Можливе використання API GPT для створення сервісу автоматичного перекладу тексту. Він може бути корисним для миттєвого перекладу текстів або створення мультязичних інтерфейсів для застосунків та вебсайтів;

- аналіз тексту, можливості аналізу тексту GPT для виявлення та витягнення важливої інформації з текстових даних. Наприклад, це може бути інструмент для автоматичного аналізу новинних статей або створення звітів на основі текстових даних;

- синтез мовлення. GPT використовується для створення системи синтезу мовлення, яка може генерувати аудіофайли з тексту. Це може бути корисним для розробки текст-на-мовлення застосунків, аудіокниг або допоможе в створенні асистентів, які взаємодіють з користувачами голосом;

- фільтрація та модерація контенту. Використання GPT для розробки інструментів фільтрації та модерації контенту, які автоматично визначають небажаний або образливий матеріал в текстах, коментарях або інших джерелах контенту;

- навчання та доповнення моделі. Ми можемо використовувати API GPT для навчання моделі на власних даних та створення власних моделей для конкретних завдань, наприклад, швидкий пошук актуальної для користувача інформації на вебсайті.

API GPT відкриває широкі можливості для створення різноманітних інтелектуальних застосунків, які можуть забезпечити значну користь для користувачів у багатьох галузях.

1.5 Постановка задачі дослідження

В цілому робота показала можливість використання мовних моделей GPT для відповідей на основі власного контенту. Застосування такого підходу для вирішення задачі консультування абітурієнтів університету в автоматичному режимі дозволить налагодити кращу проінформованість абітурієнтів та поліпшити розуміння їх потреб.

Об'єктом дослідження являється розробка і дослідження методу можливості донавчання мовної моделі ChatGPT для вирішення задачі створення нейроконсультанта для абітурієнтів.

Метою дослідження є створення нейроконсультанта для відповідей на запити абітурієнтів щодо інформації вебсайту Харківського Національного Університету Радіоелектроніки [12].

Для досягнення мети необхідно вирішити такі завдання:

- вивчити підходи щодо донавчання мовної моделі GPT на власному контенті;
- з метою зменшення обсягу власного контенту, що подається до моделі, дослідити та використати алгоритми та готові бібліотеки для обробки текстової інформації для пошуку фрагментів, що відповідають запиту користувача;
- дослідити проблему формування та налаштування промптів до GPT для формування відповіді на запит користувача;
- підготувати власний контент (базу знань), який буде використаний для відповідей абітурієнтів університету;
- розробити ядро нейроконсультанта для абітурієнтів;
- оцінити якість відповідей нейроконсультанта на основі власного датасету.

2 РОЗРОБКА ТА ДОСЛІДЖЕННЯ МЕТОДУ НАВЧАННЯ МОДЕЛІ GPT НА ВЛАСНОМУ КОНТЕНТІ

2.1 Використання GPT через API для відповідей на основі власного контенту

Модель GPT-3.5 дозволяє отримати інформацію на основі даних, які були актуальні до вересня 2021 року, GPT-4 – до квітня 2023. Тому ці моделі не володіють інформацією, що була додана до інтернету після цього. До того ж моделі GPT навряд володіють детальною інформацією щодо невеличких підприємств, шкіл, університетів, мовних курсів та інше. Наприклад, користувач бажає дізнатися таку інформацію, яка є на сайтах компаній у відкритому доступі, але модель GPT такої інформації не має, наприклад, це можуть бути вакансії щодо працевлаштування у певну компанію, умови скасування бронювання у якийсь готель, вимогами щодо рівня знань для зарахування на конкретні мовні курси, перелік лікарських препаратів компанії, які містять певний інгредієнт, та інше. Звісно користувач може знайти цю інформацію самостійно на сайті, але потрібно витратити час на пошук та на самостійну обробку великого об'єму інформації. ChatGPT позбавлений доступу до Інтернету і спирається лише на інформацію на який навчався, зате надає безпосередню відповідь на питання, що економить багато часу.

Для використання моделей GPT для вирішення конкретних задач в торгівлі, освіті, сфері послуг та інше необхідно модель доучити. Хоча компанія OpenAI вже надає доступ для класичного донавчання своїх моделей із зміною вагів нейронної мережі, в роботі буде застосовано інший підхід.

Причин цьому декілька:

- конфіденційність (конфіденційні дані компанії необхідно завантажувати на сервери OpenAI, що для деяких компаній недопустимо);

- необхідно підготувати дуже великий датасет (дані та правильні відповіді), що наразі затребує витрат матеріальних та часу від компаній;
- вартість завантаження датасету на сервери OpenAI та вартість безпосередньо самого донавчання моделі.

Тому можна використати інший підхід, а саме передати моделі GPT інформацію (базу знань), на основі якої вона повинна формувати відповідь.

У класичному сенсі під донавчанням розуміється зберігання ваг нейронної моделі для більшості шарів, та тренування останніх шарів на своєму датасеті. Але для цього необхідно мати об'ємний датасет з розміткою, формування якої потребує дуже багато часу.

Під донавчанням ChatGPT в роботі буде розумітися передача моделі власної бази знань у текстовому форматі, де буде міститися інформація, на основі якої модель буде формувати відповідь. Ця інформація буде подаватися в режимі реального часу, і донавчання моделі в класичному розумінні не потрібне. Такій підхід також дозволить подавати моделі найактуальнішу інформацію, що неможливо при використанні моделі яка навчена на застарілих даних.

Для спілкування с моделлю GPT існують такі ролі:

- system;
- user;
- assistant.

System містить інструкцію-промпт, що допомагає моделі пояснити як треба виконати завдання. User – це запит користувача, assistant – відповідь моделі. Тобто можна використати роль system для передачі моделі власного контенту (власної бази знань) з інструкцією, яка передасть моделі вимогу відповідати тільки на основі цього контенту.

Але на жаль всю базу знань подати моделі GPT не можливо. Це зв'язано з обмеженням моделі щодо кількості символів (вимірюваної в токенах), яку можна подати на вхід моделі під час обробки запиту користувача за один раз. Наприклад, GPT-3.5 підтримує токенів до 4 КБ, GPT-4 – до 8 КБ або 32 КБ. До

того ж використання через API

Щоб скоротити обсяг даних для подачі до моделі можна використати додаткові алгоритми, або готові бібліотеки, що знайдуть фрагменти, які будуть найбільш релевантними щодо запиту користувача. А потім на вхід моделі GPT вже будуть подані тільки ці фрагменти, підготовлений промпт та запит користувача. Важливо, щоб інформація, яку містять ролі system, user, assistant, сумарно не перевищувала обмеження щодо кількості токенів.

Також для скорочення обсягу тексту можна використовувати узагальнення, тобто сумаризацію (summarization). Узагальнення тексту (або саммаризація) – це процес перетворення довгого тексту на його коротку версію, зберігаючи при цьому основні ідеї та важливу інформацію оригінального тексту. Це метод часто використовують у сфері NLP. У контексті «донавчання» або тонкого налаштування моделей GPT.

2.2 Особливості використання GPT через API

Для інтеграції ChatGPT у сторонні сервіси та продукти необхідний API-ключ, який можна отримати в особистому кабінеті на сайті OpenAI [2]. Далі необхідно надіслати моделі запит, що містить ключ API та вхідні дані (питання від користувача та додаткову інструкцію-prompt, як треба моделі формувати відповідь).

Далі в роботі не буде уточнюватися версія моделі, а просто використовуватися назва модель GPT, тому запропонований підхід, застосовується до обох версій моделей ChatGPT.

Розглянемо деякі особливості використання GPT через API, які включають токени, рівні доступу (system та user), інтеграцію з іншими системами тощо:

– автентифікаційний токен (API ключ). При використанні API для GPT, зазвичай потрібен API ключ або інший тип автентифікаційного токена, який

дає доступ до сервісу;

- обмеження доступу. Токени можуть бути пов'язані з обмеженнями доступу, такими як обмеження на кількість запитів чи обмеження на типи запитів;

- системний доступ (system-level access). Деякі API можуть мати системний рівень доступу, який надає більші можливості, такі як адміністрування та управління API, створення та керування запитами, обмеженням доступу тощо;

- користувацький доступ (user-level access) – це рівень доступу, який може обмежувати можливості користувача в межах встановлених правил системи чи API.

Деякі API мають обмеження на кількість запитів, які можна здійснити протягом певного періоду часу. Це може бути у вигляді квот або лімітів на кількість запитів на день, годину, тощо. Через API, GPT може інтегруватися з іншими системами, дозволяючи їм використовувати можливості генерації тексту. Інтеграція може вимагати документації API, бібліотек або SDK для зручного використання функцій GPT у власному програмному забезпеченні.

Багато API надають документацію, приклади коду та підтримку для користувачів, що допомагає розуміти як використовувати їхні можливості.

2.3 Використання фреймворку LangChain для визначення фрагментів, що відповідають запиту користувача

В даній роботі для пошуку релевантних фрагментів запропоновано використовувати фреймворк з відкритим вихідним кодом LangChain, який реалізує велику множину задач для спрощення розроблення застосунків із використанням моделей LLM [13], таких як GPT від OpenAI.

Ключові можливості Langchain:

- Langchain дає змогу легко інтегрувати та використовувати різні

мовні моделі, зокрема GPT-3, GPT-3.5 turbo, GPT-4;

- Langchain надає інструменти для індексації великих обсягів тексту, що дає змогу швидко шукати та витягувати інформацію;
- бібліотека може бути використана для створення систем, які автоматично відповідають на запитання на основі наданого тексту.

Langchain може окрім моделей GPT від OpenAI працювати з Bidirectional Encoder Representations from Transformers (BERT) від Google та його похідними RoBERTa, DistilBERT та іншими, хоча вони переважно використовуються для розуміння мови, а не для її генерації; Text-to-Text Transfer Transformer (T5) від Google, яка орієнтована на перетворення тексту в текст, що робить її корисною для багатьох завдань NLP; XLNet та іншими трансформерами, які поліпшуються за рахунок більш складної архітектури або унікальних підходів до навчання; з локально розгорнутими моделями, що може бути корисно в ситуаціях, коли потрібен повний контроль над моделлю або коли необхідно враховувати конфіденційність даних.

LangChain Libraries є набором бібліотек для Python і JavaScript, що містять інтерфейси та інтеграції для різних компонентів. Вони надають базове середовище виконання для з'єднання цих компонентів у ланцюжки та агенти, а також включають готові реалізації цих ланцюжків і агентів [рисунок. 2.1.]. LangChain Templates – це колекція готових до використання архітектур для широкого спектра завдань. LangServe – це бібліотека для розгортання ланцюжків LangChain у вигляді REST API. LangSmith пропонує платформу для розробників, що полегшує налагодження, тестування, оцінку та моніторинг ланцюжків, створених на будь-якій платформі LLM, і забезпечує легку інтеграцію з LangChain.

Перед використанням LangChain власний контент поділяється на невеликі блоки, які спеціальним образом переводяться до embedding простору. Далі за допомогою LangChain визначаються блоки, які якнайкраще відповідають запиту користувача. Ці блоки і додаються до system, який містить також промпт, що говорить моделі GPT формувати відповідь тільки

на основі даних з цих блоків. Хочеться звернути увагу, що розмір токена визначається по різному для різних мов. Якщо спілкування з GPT йде українською мовою, то один токен це приблизно одна третина слова. Якщо текст англійською, то токен це приблизно одно слово. Тобто для збільшення кількості інформації можна використовувати переклад на англійську. Наповнення ми можемо побачити на рисунку 2.1.

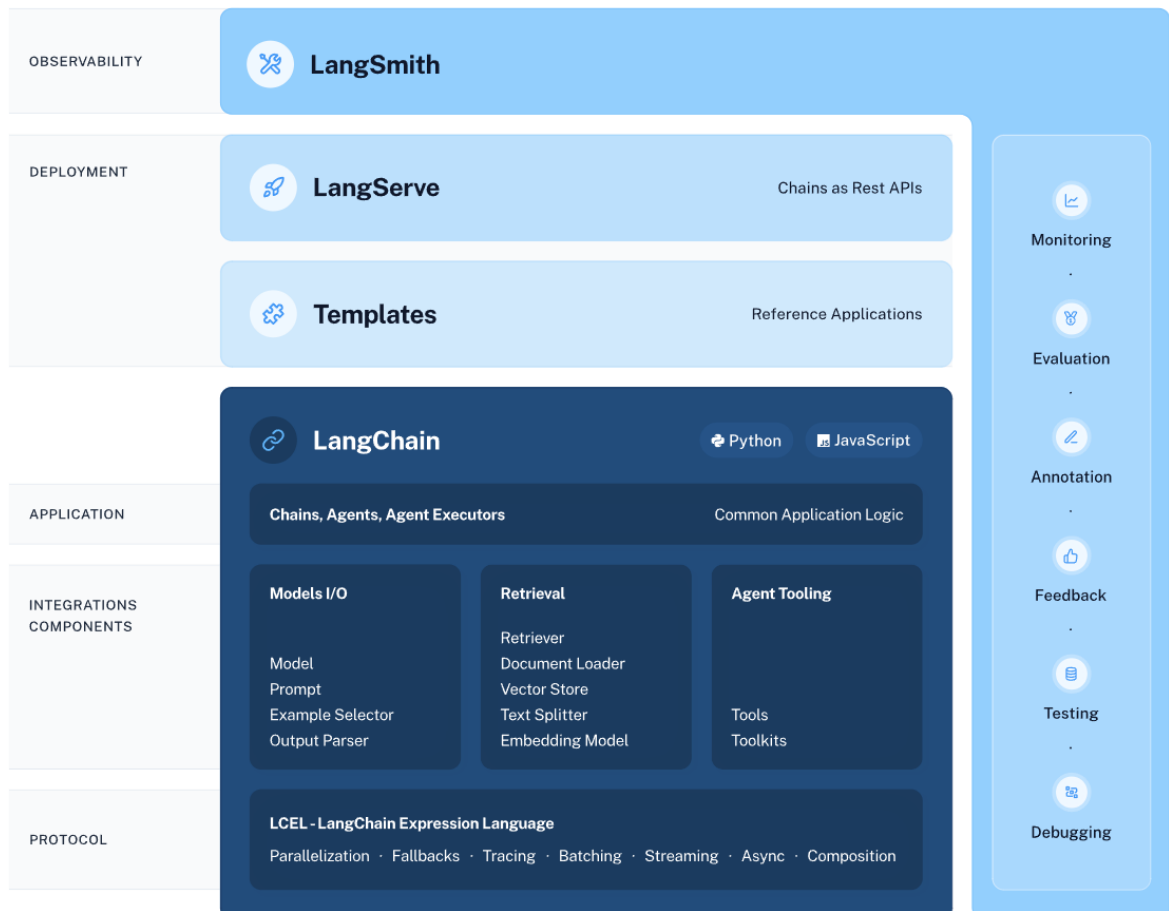


Рисунок 2.1 – Наповнення фрейворку LangChain [10]

2.4 Параметри налаштування моделі GPT інструменту LangChain щодо формування відповіді

Окрім параметру `prompt` на відповідь моделі GPT через API впливають інші параметри:

- `max_tokens` (максимальна кількість токенів);

- temperature (температура);
- top_k (sampling);
- top_p (nucleus sampling, вибір ядра);
- frequency_penalty (штраф за частоту);
- presence_penalty (штраф за присутність);
- best_of (кількість генерацій відповідей перед вибором найкращої);
- stop (Stop Sequences, зупинні послідовності);
- echo (включення промпт у відповідь);
- logprobs (логарифмічні ймовірності для вказаної кількості найімовірніших токенів);
- model (модель);
- user (користувач).

Параметр `max_tokens` визначає максимальну довжину відповіді, яку згенерує модель. Токен може бути словом, частиною слова або розділовим знаком. Збільшення цього параметра дає змогу моделі генерувати довші відповіді, або обмежити довжину відповіді. Цей параметр дає змогу контролювати довжину відповіді, що особливо важливо для застосунків, де потрібна стислість або, навпаки, докладність. Довші відповіді вимагають більше обчислювальних ресурсів і часу для обробки. Обмеження довжини відповіді може допомогти оптимізувати використання ресурсів.

Параметр `temperature` (температура) впливає на розподіл імовірностей токенів при виборі наступного слова або символу, визначає ступінь випадковості або творчості у відповідях. Низька температура (ближче до 0) змушує модель обирати найімовірніші токени, роблячи текст більш передбачуваним і менш різноманітним. Висока температура (ближче до 1, або 2) збільшує шанси менш імовірних токенів бути обраними, що призводить до більш креативних і несподіваних результатів. Дозволяє адаптувати стиль відповідей моделі залежно від завдання. Наприклад, для творчого листа можна встановити вищу температуру, а для більш інформаційних запитів – нижчу.

Цей параметр підходить для контролю загального рівня креативності та оригінальності тексту.

Параметр `top_k` (`top-k sampling`) – техніка налаштування вибірки токенів, де з усієї множини можливих наступних токенів (слів, символів) для генерації тексту обирають тільки k найімовірніших токенів. Потім наступний токен вибирають випадковим чином із цього обмеженого набору. Цей параметр дає змогу контролювати різноманітність у тексті, що генерується. Занадто велике значення k може зробити текст незв'язним або менш передбачуваним, у той час як занадто маленьке значення може призвести до надмірної передбачуваності та повторюваності. `Top-k Sampling` допомагає знайти баланс між генерацією точного та інформативного тексту й уникненням надмірного повторення або банальності.

Параметр `top_p` (`Nucleus Sampling`) – це техніка вибірки, за якої з усієї множини можливих наступних токенів (слів або символів) обирають підмножину, сумарна ймовірність яких перевищує заданий поріг P . Цей поріг зазвичай задають у вигляді десяткового дроби від 0 до 1. Замість того щоб розглядати всі можливі наступні токени, модель обмежує свій вибір тими, які найвірогідніші, доти, доки сукупна ймовірність цих токенів не досягне заданого порога P . `Top P`, на відміну від `top-k sampling`, де обирають фіксовану кількість токенів, `sampling` динамічно змінює кількість розглянутих токенів залежно від контексту. Параметр `top_p` допомагає контролювати різноманітність відповідей, запобігаючи надмірно передбачуваним або повторюваним відповідям, при цьому зберігаючи релевантність і узгодженість

Параметри `frequency_penalty` (штраф за частоту) і `presence_penalty` (штраф за присутність) допомагають зменшити повторення слів і фраз у відповідях. `frequency_penalty` зменшує ймовірність токенів, що часто повторюються, тоді як `presence_penalty` зменшує ймовірність повторного використання вже згаданих токенів.

Коли слово або фраза зустрічається в тексті, `frequency_penalty` зменшує ймовірність повторного вибору цього ж слова або фрази. Що частіше слово

з'являється, то більше знижується його ймовірність бути обраним знову. Значення штрафу зазвичай знаходиться в діапазоні від 0 (без штрафу) до 1 (максимальний штраф). Високі значення призводять до більшого зниження ймовірності повторення слів.

Параметр `presence_penalty` знижує ймовірність вибору слів або фраз, які вже з'явилися в тексті, незалежно від того, скільки разів вони були використані. Цей параметр корисний, коли необхідно запобігти повторенню слів і фраз, навіть якщо вони були згадані тільки один раз. Аналогічно `Frequency Penalty`, значення знаходиться в діапазоні від 0 до 1, де вищі значення збільшують штраф за повторення слів. Цей параметр корисний, коли необхідно запобігти повторенню слів і фраз, навіть якщо вони були згадані тільки один раз.

Параметри `frequency_penalty` і `presence_penalty` часто використовуються в поєднанні з `temperature`, `top_p` і `top_k` для досягнення найкращих результатів у генерації тексту.

Параметр `best_of` визначає кількість різних відповідей, які модель згенерує на один і той самий промпт перед тим, як вибрати найкращу з них. Цей параметр використовується для підвищення ймовірності отримання більш якісної та релевантної відповіді. Зі згенерованих відповідей обирається та, яку модель вважає найкращою, ґрунтуючись на внутрішніх критеріях оцінювання, таких як узгодженість, релевантність і повнота. Цей параметр особливо корисний у ситуаціях, де критично важливо отримати найточнішу, найповнішу або найякіснішу відповідь. Використання високого значення `best_of` може вимагати більше часу і ресурсів, оскільки моделі доведеться згенерувати і оцінити кілька відповідей. Цей параметр особливо корисний у завданнях, де якість відповіді має першорядне значення, наприклад, у юридичних, медичних або наукових додатках.

Параметр `stop` (зупиночні послідовності) використовується для вказівки точок, після досягнення яких модель має припинити генерацію тексту. Цей параметр дуже корисний для контролю довжини та структури вихідних даних.

Stop sequences – це задані послідовності символів, слів або фраз, при виявленні яких модель припиняє подальшу генерацію тексту. Це може бути що завгодно, від розділових знаків до специфічних слів або фраз. Допомагає уникати генерації небажаного або нерелевантного тексту після досягнення логічного кінця повідомлення або документа.

Параметр `echo` застосовується для включення промпта у відповідь моделі (якщо дорівнює `true`). Цей параметр може бути особливо корисним у сценаріях, де потрібно бачити як запит, так і відповідь моделі в одному повідомленні. `Echo` корисний, коли потрібно ясно бачити, на який саме запит відповідає модель, що допомагає в розумінні контексту, особливо під час обробки послідовних запитів, може бути корисним під час налагодження або логування запитів і відповідей, оскільки дає змогу легко зіставити кожен запит з його відповідною відповіддю.

Параметр `logprobs` в API моделей GPT від OpenAI використовується для отримання інформації про ймовірності (логарифмічні ймовірності) різних токенів, які модель розглядала під час генерації тексту. Ця інформація може бути корисною для аналізу і розуміння того, як модель робить свої вибори під час генерації відповідей. Параметр `logprobs` вказує, для скількох найімовірніших токенів на кожному кроці генерації ви хочете отримати логарифмічні ймовірності. Логарифмічна ймовірність – це логарифм ймовірності, який зазвичай використовується, тому що він зручніший для роботи з дуже маленькими числами, якими часто є ймовірності в моделях мови. Якщо вказати значення параметру `logprobs`, наприклад `logprobs=5`, API повертає логарифмічні ймовірності для п'яти найімовірніших токенів на кожному кроці генерації. Цю інформацію можна використати для аналізу того, які слова або фрази модель вважає найімовірнішими в даному контексті, що може допомогти в розумінні й налагодженні процесу генерації тексту.

Параметр `model` в API моделей GPT від OpenAI вказує, яку саме версію або конфігурацію моделі GPT використовувати для опрацювання вашого запиту. Цей параметр визначає, яка з доступних моделей буде генерувати

відповіді на основі вашого промпта. Model – це ідентифікатор конкретної моделі мови, такої як text-davinci-002, text-curie-001, text-babbage-001, text-ada-001, gpt-3.5-turbo, gpt-4. Кожна модель має свої особливості щодо якості, швидкості та ресурсоемності. Вибір моделі залежить від цілей і вимог. Потужніші моделі, наприклад, як серія «Davinci», краще підходять для складних завдань опрацювання мови, більш легкі моделі, такі як «Ada», можуть бути придатнішими для простіших завдань і швидше обробляють запити. Обрана модель безпосередньо впливає на якість, стиль і деталізацію генерованого тексту. Потужніші моделі зазвичай забезпечують більш точні та деталізовані відповіді.

Параметр user в API моделей GPT від OpenAI використовується для ідентифікації та відстеження запитів від конкретних користувачів. Це особливо корисно для застосунків, де потрібна персоналізація або для ситуацій, де потрібно відстежувати взаємодії окремих користувачів із моделлю. User – це ідентифікатор користувача, який ви можете передавати в API разом із запитом. Це може бути будь-який рядок, який однозначно ідентифікує користувача, наприклад, ім'я користувача, email, або внутрішній ID користувача. Коли користувач робить запит до API, можна включити параметр user з унікальним ідентифікатором для кожного користувача. Це дозволяє API відстежувати, хто робить запит. Параметр user може бути використаний для створення більш персоналізованого досвіду, наприклад, шляхом адаптації відповідей моделі до попередніх запитів користувача. Цей параметр корисний для збору статистики та аналізу взаємодій різних користувачів із моделлю, що може допомогти в поліпшенні сервісів і розумінні вподобань користувачів.

Таблиця 2.1 зберігає стислу узагальнену інформацію про розглянуті параметри API моделей GPT від OpenAI, які можна використовувати для досягнення кращих результатів у генерації тексту.

Таблиця 2.1 – Опис параметрів моделей GPT від OpenAI

Назва параметра	Можливі значення	Призначення
max_tokens	Числове значення	Визначає максимальну кількість токенів у відповіді
temperature	Число від 0 до 1	Регулює рівень випадковості або творчості у відповідях
top_k	Числове значення	Вибірка токенів з фіксованого числа k найбільш імовірних токенів.
top_p (вибірка ядра)	Десяткове число від 0 до 1	Динамічний вибір токенів, де сумарна ймовірність перевищує заданий поріг P
frequency_penalty	Число від 0 до 1	Знижує ймовірність повторення слів, що вже зустрічалися в тексті
presence_penalty	Число від 0 до 1	Знижує ймовірність повторення слів, які вже згадувалися, незалежно від частоти
best_of	Числове значення	Генерація кількох відповідей для вибору найкращої.
stop	Масив рядків	Зупиняє генерацію тексту при досягненні вказаної послідовності
logprobs	Числове значення	Повертає логарифмічні ймовірності для вказаної кількості найімовірніших токенів
echo	true/false	Включає початковий промпт в відповідь
model	Ідентифікатор моделі (наприклад, «text-davinci-002»)	Визначає, яку модель GPT використовувати для обробки запиту
user	Унікальний ідентифікатор користувача	Ідентифікує та відстежує запити від конкретного користувача

Інструмент LangChain також має досить велику кількість параметрів для налаштування роботи. В роботі був використаний параметр `verbose`, який дозволяє керувати рівнем подробиці або обсягу інформації, що надається під час виконання запитів або обробки даних. Тобто параметр `verbose` говорить LangChain наскільки детально інструмент повинен повідомляти про свою внутрішню роботу під час обробки запитів. Це може включати інформацію про те, як вибираються відповіді, які дані використовуються для обробки запиту та інші аспекти внутрішнього процесу. Докладний висновок може бути дуже корисним під час налагодження та аналізу, особливо під час розроблення складних систем обробки мови, де важливо розуміти, як модель приходить до певних висновків.

2.5 Алгоритм донавчання на власному контенті

Алгоритм донавчання моделі GPT на власному контенті даних відбувається за таким алгоритмом (рис. 2.2):

- зчитується запит користувача та записується до «user»;
- за допомогою LangChain знаходяться фрагменти з власного контенту, які якнайкраще відповідають запиту;
- налаштовуються параметри для формування відповіді (`temperature`, `verbose`)
- формується промпт для GPT;
- додаються знайдені фрагменти та промпт до «system»;
- відправляється до ChatGPT «system» та «user»;
- отримується відповідь «assistant» від ChatGPT;
- відповідь надається користувачу.

Якщо потрібно враховувати історію діалогу під час наступної відповіді для конкретного користувача, можна попередні запитання з «user» та відповіді із «assistant» узагальнити та результати узагальнення додати до «system» для

формування поточної відповіді.

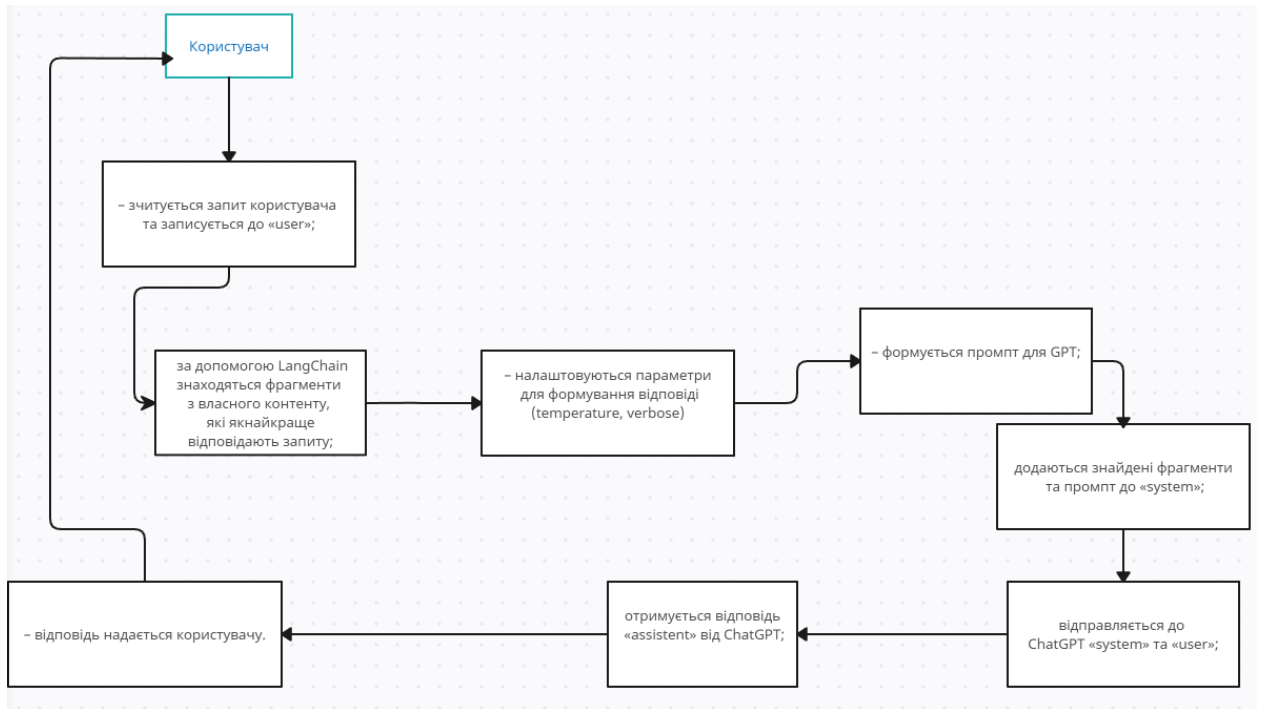


Рисунок 2.2 – Алгоритм доначення моделі GPT

3 РОЗРОБКА НЕЙРОКОНСУЛЬТАНТА ДЛЯ АБІТУРІЄНТІВ СПЕЦІАЛЬНОСТІ КОМП'ЮТЕРНІ НАУКИ, ОСВІТНЬО- ПРОФЕСІЙНА ПРОГРАМА ІНФОРМАТИКА

3.1 Налаштування програмного середовища для досліджень

Проведення дослідження буде використано мовою Python в середовищі Google Colab (Colabaratory).

Google Colab надає можливості працювати з Python у вебсередовищі з доступом до обчислювальних ресурсів Google та бібліотек, які підтримуються у Colab, що робить його зручним для досліджень та експериментів з Python.

Напочатку необхідно встановити пакети, що зображено на рисунку 3.1.

```
!pip install cohere  
!pip install openai==0.28  
!pip -q install --upgrade tiktoken  
!pip -q install langchain openai chromadb
```

Рисунок 3.1 – Встановлення пакетів

Цей розділ встановлює необхідні пакети Python (openai, tiktoken, langchain, та chromadb) для роботи з мовною моделлю OpenAI, збереженням документів, вбудовуванням, векторним сховищем, розділенням тексту та іншими функціями.

Наступним кроком імпортуємо модулі та класи що зображено на рисунку 3.2.

Завершуємо підготування до роботи з Colab імпортом та налаштуванням певних бібліотек та конфігурацій, що зображено на рисунку 3.3.

```

from langchain.llms import OpenAI
from langchain.docstore.document import Document
import requests
from langchain.embeddings.openai import OpenAIEmbeddings
from langchain.vectorstores import Chroma
from langchain.text_splitter import CharacterTextSplitter
from langchain.prompts import PromptTemplate
import pathlib
import subprocess
import tempfile
import ipywidgets as widgets
import os
import gspread
import re

```

Рисунок 3.2 – Імпортування модулів та класів

```

from langchain.llms import OpenAI
from langchain.docstore.document import Document
import requests
from langchain.embeddings.openai import OpenAIEmbeddings
from langchain.vectorstores import Chroma
from langchain.text_splitter import CharacterTextSplitter
from langchain.prompts import PromptTemplate
import pathlib
import subprocess
import tempfile
import ipywidgets as widgets
import os
import gspread
import re

```

Рисунок 3.3 – Імпорт та налаштування бібліотек та конфігурацій

На рисунку 3.4 показано формування запитів від системи щодо API-ключа OpenAI.

Цей ключ відкриває доступ до моделі GPT-3 (GPT-3.5/ChatGPT, GPT-4) через API. Його розробник отримує після реєстрації облікового запису на сайті OpenAI [10].

Також необхідно поповнити кредитний баланс за використання GPT моделей через API (рис. 3.5).

```
api_key = getpass('Введіть ваш ключ API:')
openai.api_key = api_key
```

Введіть ваш ключ API:.....

Рисунок 3.4 – Введення API-ключа від OpenAI

Також необхідно поповнити кредитний баланс за використання GPT моделей через API, що зображено на рисунку 3.5.

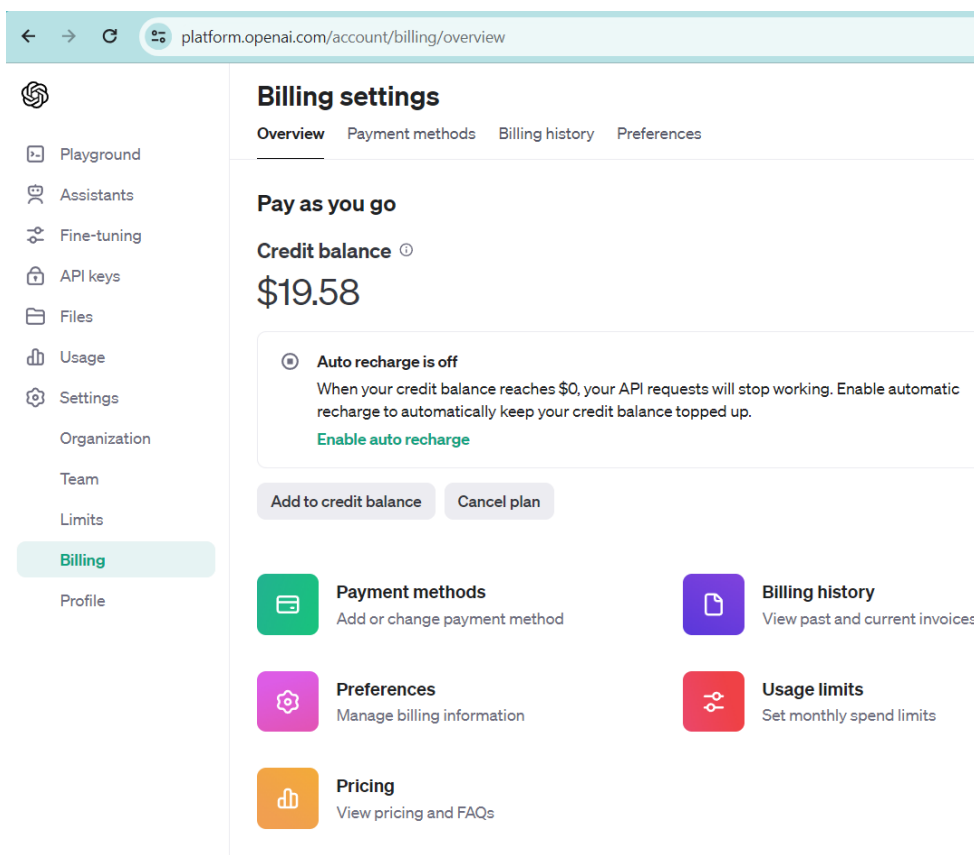


Рисунок 3.5 – Поповнення кредитного балансу за використання GPT моделей через API

Наразі все готово для того щоб почати роботу зі створення нейроконсультанта.

3.2 Підготовка власного контенту на основі інформації з сайту кафедри Інформатики

Для створення власного датасету була використана інформація з сайту кафедри Інформатики Харківського Національного Університету Радіоелектроніки [12], а саме, сторінки, де може зберігатися найбільш корисна для абітурієнта інформація:

- Про кафедру;
- про кафедру/Колектив кафедри;
- про кафедру/Міжнародна співпраця;
- про кафедру/Компанії партнери;
- наука/Напрями наукових досліджень;
- наука/Науково-дослідна робота студентів;
- наука/Навчально-наукова лабораторія;
- вступнику.

Вся інформація щодо сторінки зображена на рисунку 3.6.

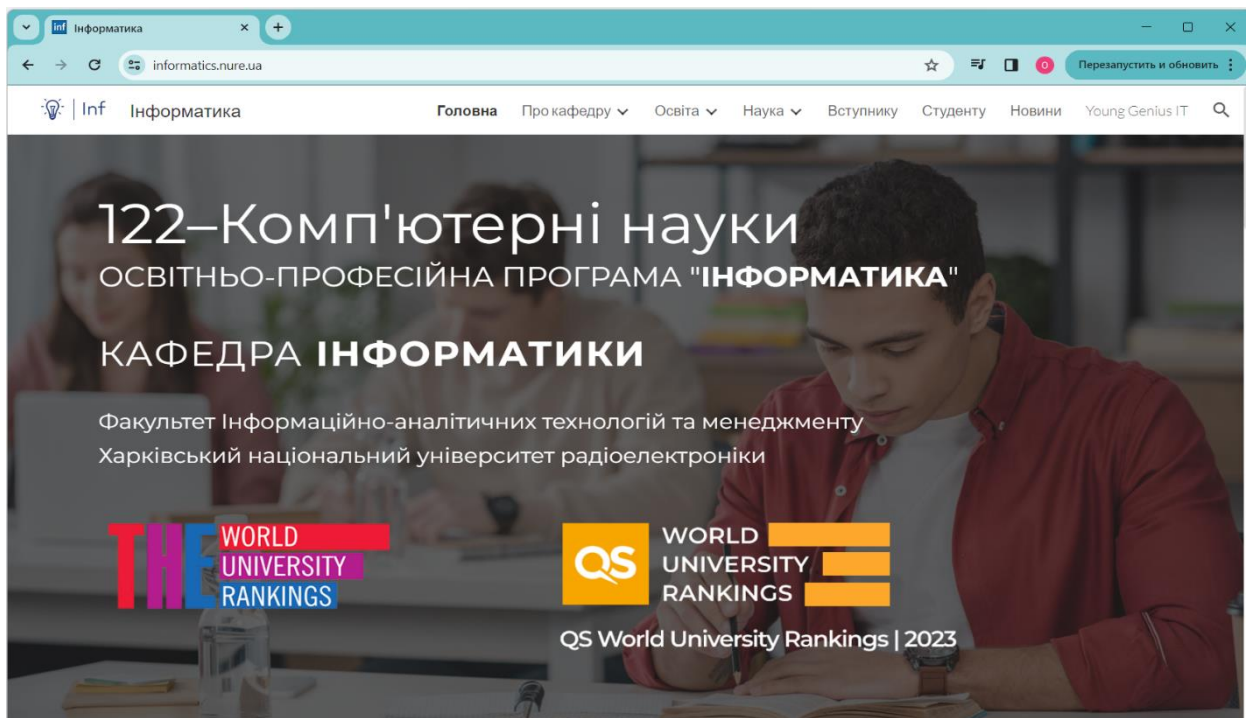


Рисунок 3.6 – Головна сторінка сайту кафедри Інформатики [12]

Інформація з наведених вище сторінок була переформатована у плоский текст і збережена у Google документі [14], фрагменти якого показовно на рисунку 3.7 та додатку Б. Таким чином, створений власний контент являє собою Google документ, що містить 35 сторінок текстової інформації.

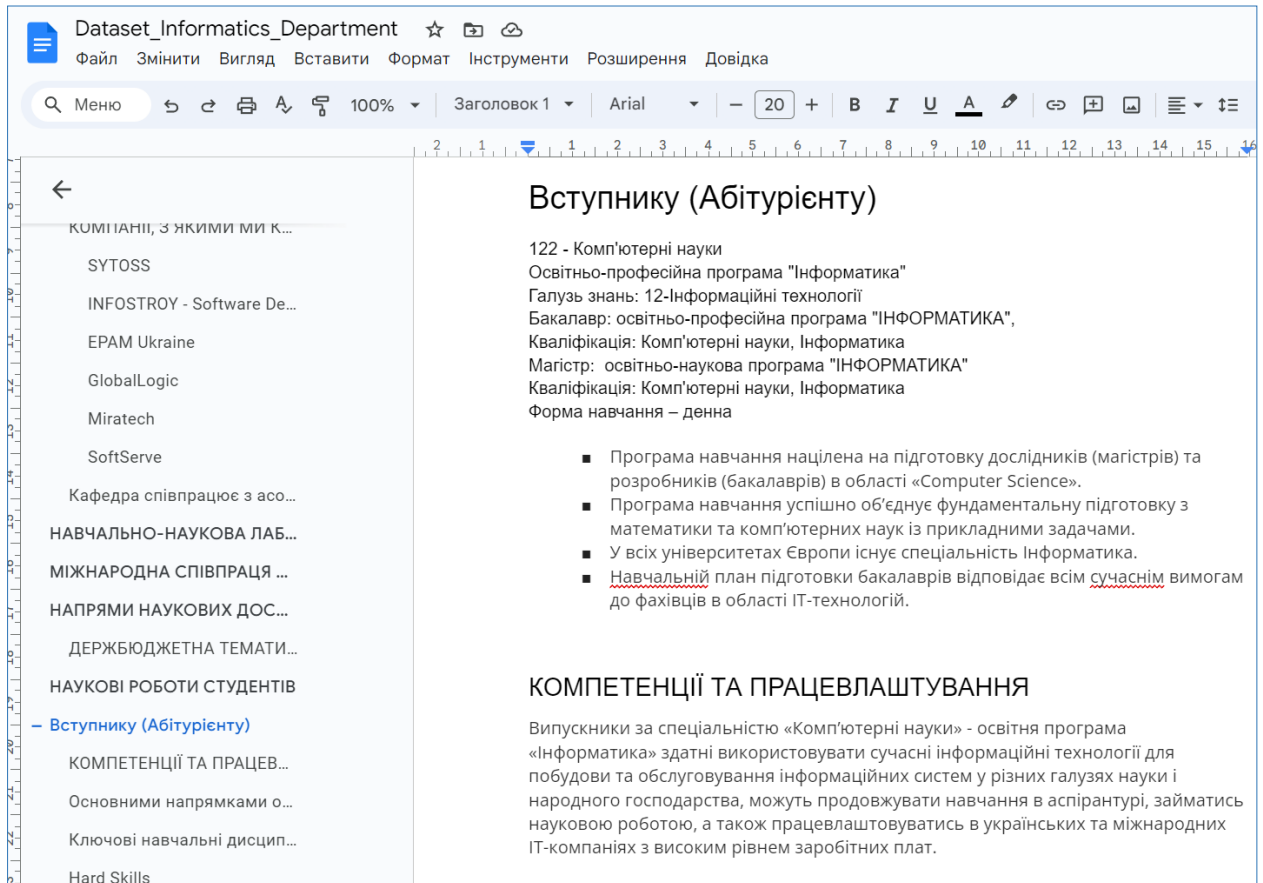


Рисунок 3.6 – Фрагмент Google документу, що містить контент для «донавчання» моделі GPT

3.3 Реалізація розробленого алгоритму щодо створення нейроконсультанту

Для розробки та тестування нейроконсультанту, який буде відповідати користувачу на основі контенту з розділу 3.2 було створенні декілька допоміжних функцій:

- `load_document_text` для завантаження тестової інформації за URL-адресою, що зображено на рисунку 3.7;
- `create_search_index` для створення пошукового індексу з тексту, що зображено на рисунку 3.8;
- `num_tokens_from_messages` для розрахунку кількості токенів у повідомленнях, використовуючи специфічне кодування для визначеної моделі (для визначення витрат на використання моделі, що зображено на рисунку 3.9);
- `create_embedding` для генерації векторних відображень (ембедінгів) для тексту, розбиває текст на фрагменти і створює пошуковий індекс з цих фрагментів, що зображено на рисунку 3.10;
- `num_tokens_from_messages` для визначає кількість токенів у списку повідомлень `messages` для заданої моделі що зображено на рисунку 3.11);
- `insert_newlines` для форматування тексту, вставляє переноси рядків для обмеження довжини рядка, щоб полегшити читання що зображено на рисунку 3.12;
- `answer_index` використовує пошуковий індекс для знаходження релевантних документів до заданого питання і генерує відповідь моделі GPT на це питання що зображено на рисунку 3.13;
- `answer_user_question` для завантаження промпта, бази знань (з заданих URL), запиту користувача, та генерації відповіді на питання користувача що зображено на рисунку 3.14);

Далі наведено функції більш детально.

Функція `load_document_text` (рисунок 3.7) виконує такі дії:

- перевірка URL: спочатку код перевіряє, чи введений URL дійсно є посиланням на Google Docs. Це робиться за допомогою регулярного виразу, який шукає специфічний шаблон `/document/d/([a-zA-Z0-9-_]+)` у URL;
- вилучення ID документа: якщо URL відповідає шаблону, код витягує ідентифікатор документа з URL;
- завантаження тексту: за допомогою HTTP-запиту до Google Docs код

завантажує текст документа у форматі .txt за допомогою вилученого ідентифікатора документа;

– повернення тексту: текст документа повертається як результат функції.

```
def load_document_text(url: str) -> str:
    match_ = re.search('/document/d/([a-zA-Z0-9-_-]+)', url)
    if match_ is None:
        raise ValueError('Invalid Google Docs URL')
    doc_id = match_.group(1)

    response = requests.get(f'https://docs.google.com/document/d/{doc_id}/export?format=txt')
    response.raise_for_status()
    text = response.text

    return text
```

Рисунок 3.7 – Функція load_document_text для завантаження тестової інформації за URL

Функція create_search_index призначена для створення пошукового індексу на основі заданого тексту. Вона викликає іншу функцію create_embedding, яка, генерує векторні представлення (ембедінги) для цього тексту (рис.Х.2). Ось її детальний опис:

– параметр входу: функція приймає рядок тексту як вхідний параметр;

– генерація ембедінгів: у функції create_embedding текст перетворюється на векторні представлення, які можуть використовуватися для пошуку та аналізу текстових даних;

– повернення індексу: функція повертає об'єкт Chroma, який представляє собою структуру даних, що дозволяє ефективно проводити пошук по тексту.

Ця функція є ключовою для створення структури, яка дозволить швидко і точно знаходити інформацію у великому обсязі тексту.

```
def create_search_index(text: str) -> Chroma:
    return create_embedding(text)
```

Рисунок 3.8 – Функція `create_search_index` для створення пошукового індексу з тексту

Функція `num_tokens_from_string` призначена для підрахунку кількості токенів у тексті (рисунок.3.9):

- приймає два параметри:
- `string`: рядок тексту, в якому потрібно підрахувати токени,
- `encoding_name`: назва кодування, яке використовується для токенизації тексту;
- отримання кодування для токенизації: за допомогою `tiktoken.get_encoding(encoding_name)` отримує метод токенизації на основі заданого імені кодування;
- кодування та підрахунок токенів: використовує метод `encode` від отриманого кодування для перетворення рядка `string` на токени та підраховує їх кількість;
- повертає кількість токенів: Функція повертає ціле число, яке представляє кількість токенів у заданому тексті.

Ця функція корисна в контексті обробки природної мови або для визначення обсягу тексту у термінах токенів, що є важливим для моделей машинного навчання, зокрема у контексті мовних моделей, які мають обмеження на максимальну кількість токенів.

```
def num_tokens_from_string(string: str, encoding_name: str) -> int:
    """Returns the number of tokens in a text string."""
    encoding = tiktoken.get_encoding(encoding_name)
    num_tokens = len(encoding.encode(string))
    return num_tokens
```

Рисунок 3.9 – Функція `num_tokens_from_string` призначена для підрахунку кількості токенів у тексті

Функція `create_embedding` призначена для створення пошукового

індексу з набору текстових даних (рис.3.10):

- розділення тексту на частини (чанки, chunks): використовує `CharacterTextSplitter` для розділення вхідного тексту (`data`) на фрагменти, виходячи з розміру частини та роздільника (використано: символ переносу рядка `separator="\n"`, розмір фрагменту `chunk_size=1024`).

- створення об'єктів `Document`: кожен фрагмент тексту перетворюється на об'єкт `Document`, який зберігає вміст сторінки і метадані.

- створення пошукового індексу `Chroma`. Використовує `Chroma.from_documents` для створення пошукового індексу з масиву об'єктів `Document`. Індекс створюється за допомогою функції `OpenAIEmbeddings` з бібліотеки `LangChain` (`from langchain.embeddings.openai import OpenAIEmbeddings`). Цей індекс забезпечує можливість ефективного пошуку за текстовими даними;

- підрахунок токенів: використовує `num_tokens_from_string` для підрахунку загальної кількості токенів у всіх фрагментах тексту.

- повернення пошукового індексу: функція повертає створений пошуковий індекс `Chroma`.

```
def create_embedding(data):
    source_chunks = []
    splitter = CharacterTextSplitter(separator="\n", chunk_size=1024, chunk_overlap=0)

    for chunk in splitter.split_text(data):
        source_chunks.append(Document(page_content=chunk, metadata={}))

    search_index = Chroma.from_documents(source_chunks, OpenAIEmbeddings(openai_api_key=api_key), )

    count_token = num_tokens_from_string(' '.join([x.page_content for x in source_chunks]), "cl100k_base")

    return search_index
```

Рисунок 3.10 – Функція `create_embedding` для створення пошукового індексу з набору текстових даних

Функція `num_tokens_from_messages` визначає кількість токенів у списку повідомлень `messages` для заданої моделі GPT-3.5-turbo (рис. 3.11):

– спочатку вона намагається отримати кодування для моделі за допомогою `tiktoken.encoding_for_model(model)`. У разі виникнення помилки `KeyError` використовується стандартне кодування `cl100k_base`;

– для моделі «gpt-3.5-turbo-0301» вона ітерує через повідомлення, додаючи до загальної кількості токенів 4 токени за кожне повідомлення, а потім додає кількість токенів, яка відповідає довжині закодованого вмісту кожного повідомлення;

– якщо ключ у повідомленні є «name», від загальної кількості токенів віднімається 1; після обробки всіх повідомлень до загальної кількості токенів додається 2.

Функція повертає загальну кількість токенів. У разі, якщо модель не є «gpt-3.5-turbo-0301», функція викликає виняток `NotImplementedError`.

```
def num_tokens_from_messages(messages, model="gpt-3.5-turbo-0301"):
    try:
        encoding = tiktoken.encoding_for_model(model)
    except KeyError:
        encoding = tiktoken.get_encoding("cl100k_base")
    if model == "gpt-3.5-turbo-0301":
        num_tokens = 0
        for message in messages:
            num_tokens += 4
            for key, value in message.items():
                num_tokens += len(encoding.encode(value))
                if key == "name":
                    num_tokens += -1
            num_tokens += 2
        return num_tokens
    else:
        raise NotImplementedError(f"num_tokens_from_messages() is\
not presently implemented for model {model}.\
See https://github.com/openai/openai-python/blob/main/chatml.md \
for information on how messages are converted to tokens.")
```

Рисунок 3.11 – Функція `num_tokens_from_messages` для визначення кількості токенів у списку повідомлень `messages`

Функція `insert_newlines` призначена для форматування тексту, додаючи переноси рядків таким чином, щоб жоден рядок не перевищував задану

максимальну довжину (рис. 3.12):

- розбиває вхідний текст `text` на окремі слова;
- ітерує через слова, додаючи їх до поточного рядка `current_line`;
- якщо додавання нового слова перевищує максимальну довжину рядка `max_len`, поточний рядок додається до списку `lines`, а `current_line` очищається;
- після проходження всіх слів, останній рядок також додається до `lines`.

Функція повертає всі рядки, об'єднані з переносами рядків. Ця функція корисна для переформатування тексту перед його виведенням або друкуванням, щоб гарантувати, що жоден рядок не буде занадто довгим.

```
def insert_newlines(text: str, max_len: int = 170) -> str:
    words = text.split()
    lines = []
    current_line = ""
    for word in words:
        if len(current_line + " " + word) > max_len:
            lines.append(current_line)
            current_line = ""
        current_line += " " + word
    lines.append(current_line)
    return "\n".join(lines)
```

Рисунок 3.12 – Функція `insert_newlines` для форматування тексту

Функція `answer_index` використовується для генерації відповідей на запитання, базуючись на пошуковому індексі та використовуючи модель GPT-3.5-turbo (рис.3.13):

- пошук схожих фрагментів, схожих на задану тему `topic` у `search_index`, вибираючи п'ять найбільш релевантних;
- формування вмісту повідомлення: якщо параметр `verbose` активний, виводить роздільні лінії та вміст повідомлення, що включає отримані фрагменти;
- формує список `messages`, де включені два типи повідомлень – одне від «user» та одне від `system`, який містить промпт та знайденні фрагменти;
- викликає модель GPT-3.5-turbo з цими `messages` та заданим

параметром `temperature` для генерації відповіді;

– повернення відповіді: отримує та форматує відповідь моделі, додаючи переноси рядків за потреби, і повертає її користувачеві.

```
def answer_index(system, topic, search_index, temp=1, verbose=0):

    docs = search_index.similarity_search(topic, k=5)
    if verbose: print('\n =====: ')
    message_content = re.sub(r'\n{2}', ' ', '\n '.join\
        ([f'\nОтрывок документа №{i+1}\n===== ' + doc.page_content + '\n' for i, doc in enumerate(docs)])
    if verbose: print('message_content :\n ===== \n', message_content)

    messages = [
        {"role": "system", "content": system + f"{message_content}"},
        {"role": "user", "content": topic}
    ]

    if verbose: print('\n =====: ')
    if verbose: print(f"{num_tokens_from_messages(messages, 'gpt-3.5-turbo-0301')} tokens used for the question")

    completion = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=messages,
        temperature=temp
    )

    if verbose: print('\n =====: ')
    if verbose: print(f"{completion[\"usage\"][\"total_tokens\"]} total tokens used (question-answer).")
    if verbose: print('\n =====: ')
    answer = insert_newlines(completion.choices[0].message.content)

    return answer
```

Рисунок 3.13 – Функція `answer_index` для генерації відповідей на запитання, базуючись на пошуковому індексі

Функція `answer_user_questionNew` призначена для отримання відповіді на питання користувача, використовуючи текст з промпту та бази знань (рис.3.14):

– завантаження текст промпту з URL системної документації (`system_doc_url`) та з URL-адреси бази знань (`knowledge_base_url`) за допомогою функції `load_document_text`;

– створення пошукового індексу: використовує функцію `create_search_index` для створення пошукового індексу з тексту бази знань;

– формування запиту користувача;

- отримання відповіді на питання користувача за допомогою функції `answer_index`, використовуючи текст промпту, пошуковий індекс бази знань та тест питання користувача;
- повернення відповіді.

```
def answer_user_questionNew(system_doc: str, knowledge_base_url: str, user_question: str) -> str:
    system_doc_text = load_document_text(system_doc_url)
    knowledge_base_text = load_document_text(knowledge_base_url)
    knowledge_base_index = create_search_index(knowledge_base_text)
    input_text = user_question

    answer_text = answer_index(system_doc_text, input_text, knowledge_base_index, \
                               temp=temperature, verbose=verbose)

    return answer_text
```

Рисунок 3.14 – Функція `answer_user_questionNew` призначена для отримання відповіді на питання користувача на основі тексту з промпту та бази знань

3.4 Формування промтів та налаштування параметрів моделі GPT та LangChain

Промпт (prompt) для моделі GPT – це вхідні дані або запит, який користувач подає моделі. Промпт може бути текстовим повідомленням, запитанням, твердженням або навіть набором слів, який дає моделі початкову точку для генерації відповіді або продовження тексту. У разі GPT, модель використовує промпт для генерації тексту, який відповідає заданій темі, стилю або контексту.

Як наведено в розд. 2.1 промпт може зберігати як інструкцію для формування відповіді, так і підготовленні LangChain фрагменти власного контенту, що якнайкраще відповідають запиту користувача.

Для пошуку схожих фрагментів було створена функція `create_search_index` (рис. 3.8), яка використовує функцію `OpenAIEmbeddings` з

бібліотеки LangChain. Функція `create_search_index` проходить через весь наданий текст (або тексти) і створює індекс. Після створення індексу, під час надходження запиту від користувача, цей індекс використовується для швидкого пошуку та вилучення найбільш релевантних фрагментів тексту. В роботі було вирішено весь контент поділяти на блоки 1024 символів та відбирати 5 найбільш релевантних фрагментів.

Були досліджені декілька варіантів інструкцій-промптів для максимально точної відповіді моделі на запит абітурієнта. Як найкращий було обрано промпт, наведений на рисунку 3.15. Такий промпт сберігався та подавався системі теж як Google документ [15].

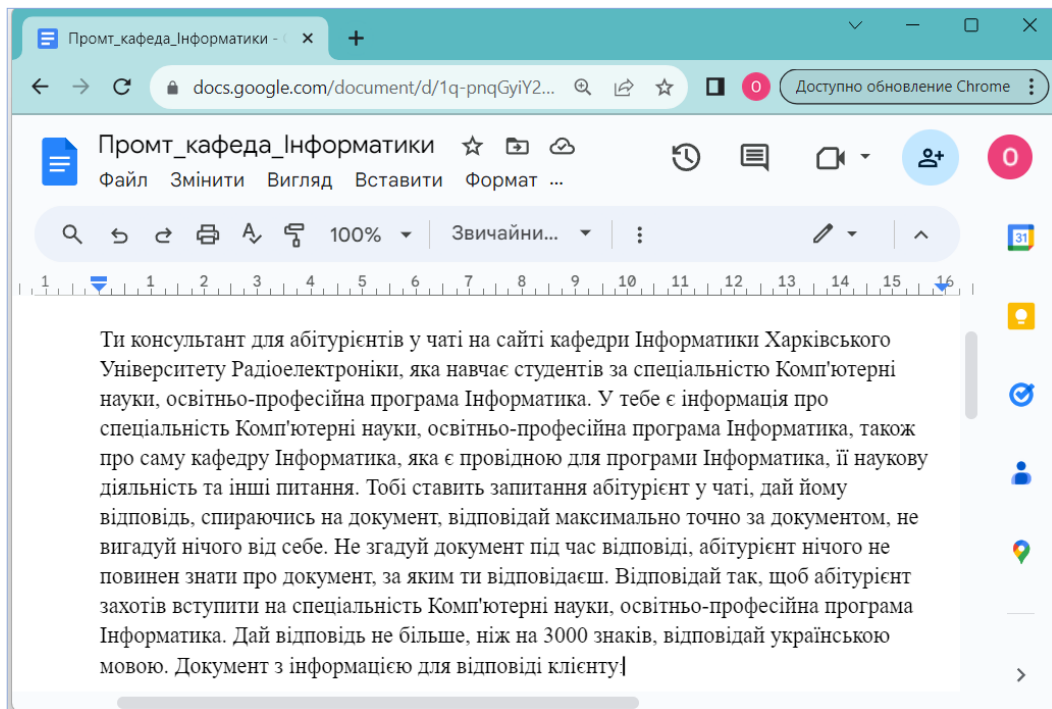


Рисунок 3.15 – Інструкція-промпт, що подавалася з обраними за допомогою бібліотеки LangChain фрагментами до «system»

В роботі були проведено експерименти з різними значення параметрів `temperature`, `verbose`.

На рисунках 3.16 – 3.25 показаний результат роботи системи з використанням різних значень параметру `temperature`, який відповідає за рівень випадковості або творчості у відповідях. Так на рисунку 3.16 показано

результат для експерименту, де модель давала відповідь на запитання «Чому потрібно обрати освітню програму Інформатика, які вона має переваги перед іншими програмами, пов'язаними з ІТ?», де параметр `temperature` дорівнював значенню 0. Декілька запусків показали, що результат змінювався не суттєво. Далі були проведені експерименти, де параметр `temperature` дорівнював значенню 1.

```
temperature=0
verbose=0
# вопрос пользователя
topic='Чому потрібно обрати освітню програму Інформатика, які вона має\
      \переваги перед іншими програмами, пов'язаними з ІТ?'
ans=answer_user_questionNew(system_doc_url, knowledge_base_url, topic)
ans

' Освітня програма Інформатика має кілька переваг перед іншими програмами, пов'язаними з І
Т. Основні переваги цієї програми включають: 1. Широкий спектр знань: Освітня\n програма Ін
форматика надає студентам широкий спектр знань у сфері сучасних технологій та мов програмув
ання, розробки інформаційних та інтелектуальних систем, а також\n математичного та комп'юте
рного моделювання процесів і систем різної природи. 2. Поєднання математичної підготовки з
інформаційними технологіями: Особлива увага\n приділяється поєднанню математичної підготовк
и студентів зі знаннями в області інформаційних технологій. Це дозволяє випускникам програм
и Інформатика бути компетентними\n в розробці та аналізі складних багатовимірних даних. 3.
Високий рівень заробітних плат: Випускники програми Інформатика мають можливість працевлашт
овуватись в\n українських та міжнародних ІТ-компаніях з високим рівнем заробітних плат. Це
забезпечує випускникам хороші перспективи для кар'єрного росту та фінансового успіху. 4.\n
М...
```

Рисунок 3.16 – Експеримент 1, запуск 1, `temperature=0`

```
temperature=0
verbose=0
# вопрос пользователя
topic='Чому потрібно обрати освітню програму Інформатика, які вона має\
      \переваги перед іншими програмами, пов'язаними з ІТ?'
ans=answer_user_questionNew(system_doc_url, knowledge_base_url, topic)
ans

' Освітня програма Інформатика має кілька переваг перед іншими програмами, пов'язаними з І
Т. Основні переваги цієї програми включають: 1. Широкий спектр знань: Освітня\n програма Ін
форматика надає студентам широкий спектр знань у сфері сучасних технологій та мов програмув
ання, розробки інформаційних та інтелектуальних систем, а також\n математичного та комп'юте
рного моделювання процесів і систем різної природи. 2. Поєднання математичної підготовки з
інформаційними технологіями: Особлива увага\n приділяється поєднанню математичної підготовк
и студентів зі знаннями в області інформаційних технологій. Це дозволяє випускникам програм
и Інформатика бути компетентними\n в розробці та аналізі складних багатовимірних даних. 3.
Високий рівень заробітних плат: Випускники програми Інформатика мають можливість працевлашт
овуватись в\n українських та міжнародних ІТ-компаніях з високим рівнем заробітних плат. Це
забезпечує випускникам хороші перспективи для кар'єрного росту та фінансового успіху. 4.\n
М...
```

Рисунок 3.17 – Експеримент 1, запуск 2, `temperature=0`

```

temperature=1
verbose=0
# вопрос пользователя
topic='Чому потрібно обрати освітню програму Інформатика, які вона має\
      \переваги перед іншими програмами, пов'язаними з IT?'
ans=answer_user_questionNew(system_doc_url, knowledge_base_url, topic)
ans

' Освітня програма "Інформатика" за спеціальністю "Комп'ютерні науки" має декілька переваг
перед іншими програмами, пов'язаними з IT. Основні з них: 1. Сучасні технології\
та мови програмування: Освіта в рамках цієї програми надає студентам глибокі знання про сучасні тех
нології та мови програмування, які є ключовими для розробки сучасних\
програмих рішень.
2. Розробка інформаційних та інтелектуальних систем: У рамках програми студенти отримують н
авички розробки інформаційних та інтелектуальних систем, що\
дозволяє їм створювати нові т
ехнологічні рішення для різних галузей. 3. Математичне та комп'ютерне моделювання процесів
і систем: Програма забезпечує студентів глибокими\
знаннями в математиці та комп'ютерному
моделюванні, що дозволяє їм розуміти та прогнозувати поведінку систем, що піддаються моделю
ванню. 4. Поєднання математичної\
підготовки з інформаційними технологіями: Освітня програ
ма надає студентам можливість поєднувати глибокі знання в математиці з професійними навич...
'

```

Рисунок 3.18 – Експеримент 1, запуск 1, temperature=1

```

temperature=1
verbose=0
# вопрос пользователя
topic='Чому потрібно обрати освітню програму Інформатика, які вона має\
      \переваги перед іншими програмами, пов'язаними з IT?'
ans=answer_user_questionNew(system_doc_url, knowledge_base_url, topic)
ans

' Освітня програма Інформатика має кілька переваг перед іншими програмами, пов'язаними з І
Т. По-перше, вона розроблена з огляду на сучасні технології та мови\
програмування, що доз
воляє студентам отримати актуальні знання і навички, відповідні вимогам ринку праці. По-дру
ге, освітня програма Інформатика надає студентам\
можливість розробляти інформаційні та ін
телектуальні системи, що є важливим напрямком розвитку сучасної науки та технологій. Це доз
воляє випускникам спеціальності\
реалізовувати свої ідеї та розвивати нові продукти та сер
віси. По-третє, освітня програма Інформатика включає математичне та комп'ютерне моделювання
процесів і систем\
різної природи. Це дозволяє студентам вивчати та розуміти складні проце
си та вирішувати проблеми, що виникають у різних галузях науки і народного господарства. Кр
ім\
того, студенти освітньої програми Інформатика отримують підготовку з обробки і аналізу
складних багатовимірних даних, що стає все більш важливим в сучасному світі. Це\
відкрива
'

```

Рисунок 3.19 – Експеримент 1, запуск 2, temperature=1

```

temperature=0
verbose=0
# вопрос пользователя
topic='Яка вартість навчання на контрактній основі на програмі Інформатика у 2023-2024 році?'
ans=answer_user_questionNew(system_doc_url, knowledge_base_url, topic)
ans

```

' Вартість навчання на контрактній основі на програмі Інформатика у 2023-2024 році становить 31900 грн.'

Рисунок 3.20 – Експеримент 2, запуск 1, temperature=0

```

temperature=0
verbose=0
# вопрос пользователя
topic='Яка вартість навчання на контрактній основі на програмі Інформатика у 2023-2024 році?'
ans=answer_user_questionNew(system_doc_url, knowledge_base_url, topic)
ans

```

' Вартість навчання на контрактній основі на програмі Інформатика у 2023-2024 році становить 31900 грн.'

Рисунок 3.21 – Експеримент 2, запуск 2, temperature=0

```

temperature=1
verbose=0
# вопрос пользователя
topic='Яка вартість навчання на контрактній основі на програмі Інформатика у 2023-2024 році?'
ans=answer_user_questionNew(system_doc_url, knowledge_base_url, topic)
ans

```

' Вартість навчання на контрактній основі на програмі Інформатика у 2023-2024 році становитиме 31900 грн.'

Рисунок 3.22 – Експеримент 2, запуск 1, temperature=1

```

temperature=1
verbose=0
# вопрос пользователя
topic='Яка вартість навчання на контрактній основі на програмі Інформатика у 2023-2024 році?'
ans=answer_user_questionNew(system_doc_url, knowledge_base_url, topic)
ans

```

' Вартість навчання на контрактній основі на програмі Інформатика у 2023-2024 навчальному році становить 31900 грн.'

Рисунок 3.23 – Експеримент 2, запуск 2, temperature=1

```

temperature=1
verbose=0
# вопрос пользователя
topic='Яка вартість навчання на контрактній основі на програмі Інформатика у 2023-2024 році?'
ans=answer_user_questionNew(system_doc_url, knowledge_base_url, topic)
ans
' Вартість навчання на програмі Інформатика на контрактній основі у 2023-2024 році становить 31900 грн.'

```

Рисунок 3.24 – Експеримент 2, запуск 3, temperature=1

Також був досліджений вплив параметру `verbose` на форму відповіді. Якщо параметр `verbose` дорівнює 0, користувач отримує тільки відповідь. Якщо `verbose=1`, також виводяться фрагменти з бази знань (`chunk`), які `LangChain` обрав, як більш релевантні (рис. 3.25).

Експерименти з параметром `temperature` показали що, коли модель дає коротку відповідь на конкретні питання, наприклад, про вартість навчання, або номеру курсу, на якому викладається певна дисципліна, і параметр `temperature=1`, то варіативність відповіді не суттєва.

Однак, якщо відповідь потребує надати більш інформації, особливо, якщо потрібне узагальнення, то варіативність суттєва, до того ж відповіді мають різні стилістику.

Далі проілюстрована роботи розробленого нейроконсультату на основі моделі GPT-3.5-turbo під час відповіді на різні питання.

Було проведено експерименти щодо відповідей на інші запитання з різними значеннями параметру `temperature`. Більше експериментів з параметром `temperature` у Додатку В.

```

+ Код + Текст

1 курс:
Об'єктно-орієнтоване програмування
Дискретна математика
Математичний аналіз
2 курс:
Бази даних та інформаційні системи
Кросплатформне програмування
Web-технології та web-дизайн
3 курс:
Інтелектуальний аналіз даних
Програмування JavaScript
Нереляційні бази даних NoSQL
Машинне навчання
4 курс:
Технології розподілених систем
та паралельних обчислень
Обробка зображень та мультимедіа
Методи аналізу зображень та відео
* Hard Skills

Database Management
Statistics & Probability
Optimization Methods
Programming
Computer Vision
Data Visualizations

* Професії випускника ОП:
Програміст-розробник
Бізнес-аналітик
Тестувальник ПЗ
Менеджер IT-проектів
Системний аналітик
ВАРТІСТЬ НАВЧАННЯ 2023-2024 РОКУ – 31900 грн.

Чому треба обирати Інформатику?
* професійний викладацький склад;
* викладання на високому рівні всіх актуальних мов і технологій, створення програмних застосунків;

=====
3088 tokens used for the question

=====
3138 total tokens used (question-answer).

=====
' Вартість навчання на контрактній основі на програмі Інформатика в 2023-2024 році становить 31900 грн.'

```

Рисунок 3.25 – Вивод одного з релевантних фрагментів, знайдений за допомогою LangChain

3.5 Ілюстрація роботи нейроконсультанта

Далі проілюстрована роботи розробленого нейроконсультату на основі моделі GPT-3.5-turbo під час відповіді на різні питання.

Питання:

– якими науковими дослідженнями займаються на кафедрі Інформатики?

Результат опрацьованої відповіді зображено на рисунку 3.26.

```
# запит користувача
topic='Якими науковими дослідженнями займаються на кафедрі Інформатики'
ans=answer_user_question(system_doc_url, knowledge_base_url, topic)
ans

' На кафедрі інформатики Харківського Університету Радіоелектроніки проводяться наукові проекти в різних областях інформатики, таких як штучний інтелект, обробка природного мови, комп'ютерне зоріння, біоінформатика і інші. Одним з наукових проектів є розробка алгоритмів машинного навчання для аналізу і класифікації великих об'ємів даних. Цей проект спрямований на створення інноваційних рішень в області аналізу даних і їх застосування в різних сферах, таких як медицина, фінанси, транспорт і інші. Участво в таких проектах дозволяє студентам розширити свої знання і навички в області інформатики, а також отримати практичний досвід роботи з сучасними технологіями.'
```

Рисунок 3.26 – Відповідь GPT-3.5-turbo на задане питання на основі контенту [14]

Питання:

– чи допомагає кафедра інформатики студентам з працевлаштуванням на основі контенту [7]. Відповідь на це питання зображено на рисунку 3.27.

```
# запит користувача
topic='Чи допомагає кафедра інформатики студентам з працевлаштуванням?'
ans=answer_user_question(system_doc_url, knowledge_base_url, topic)
ans

' Так, кафедра інформатики активно співпрацює з роботодавцями та забезпечує випускників можливостями для працевлаштування. Згідно з даними, наданими керівництвом кафедри, 100% випускників кафедри працевлаштовані за фахом. Деякі студенти навіть знаходять роботу в провідних ІТ компаніях ще з 4 курсу. Така успішна працевлаштування студентів свідчить про високу якість підготовки на кафедрі інформатики.'
```

Рисунок 3.27 – Відповідь GPT-3.5-turbo на задане питання на основі контенту [14]

Питання:

– у яких ІТ компаніях є можливість пройти стажування студентам? Ми можемо побачити відповідь чат-боту на рисунку 3.28.

```
# запит користувача
topic='у яких ІТ компаніях можна пройти стажування студентам'
ans=answer_user_question(system_doc_url, knowledge_base_url, topic)
ans

' Студенти кафедри інформатики мають можливість пройти стажування в таких ІТ компаніях: 1. EPAM Ukraine - найбільша ІТ-компанія в Україні, яка пропонує безкоштовні навчальні програми по всіх найпопулярніших технологіях і напрямках для студентів. В рамках співробітництва з кафедрою інформатики, EPAM University відкрив навчальний клас. 2. SYTOSS - компанія Full-Cycle Custom Software Development. Вона спеціалізується на створенні складних програмно-технічних рішень та обслуговуванні провідних гігантів та малого бізнесу з 2010 року. 3. INFOSTROY - провідна ІТ-компанія, яка пропонує рішення в галузі програмування та розробки програмного забезпечення. 4. GlobalLogic - компанія, яка спеціалізується на розробці програмного забезпечення та надає послуги в галузі ІТ-консалтингу. 5. Miratech - компанія, яка надає послуги в галузі програмування та розробки програмного забезпечення. 6. SoftServe - провідна ІТ-компанія, яка пропонує рішення в галузі програмування та розробки програмного забезпечення. Ці компанії співпрацюють з кафедрою інформатики і надають можливість студентам пройти стажування для практичного отримання досвіду в роботі з сучасними технологіями та отриманням нових знань.'
```

Рисунок 3.28 – Відповідь GPT-3.5-turbo на задане питання щодо стажування та працевлаштування на основі контенту [14]

3.6 Оцінка роботи нейроконсультанта групою експертів

Для оцінки якості відповідей розробленого нейроконсультанта були запрошені студенти молодших курсів, які сформуливали запитання нейроконсультанту, а також експерти, які оцінювали відповідність відповідей моделі за 5-ти бальною шкалою (від -2 до 2). У якості експертів виступили працівники кафедри Інформатики та студенти старших курсів. Низькі оцінки зазвичай були зумовлені ставленням питань, не пов'язаних з університетом, таких як запитання про погоду або наявність морозива в університеті, а також відсутністю чи недостатністю інформації в датасеті.

Це вказує на потребу подальшого розширення та удосконалення датасету.

ВИСНОВКИ

У рамках кваліфікаційної роботи був розроблений і реалізований нейроконсультант на основі моделі ChatGPT 3.5-turbo.

Був створений датасет (власний контент), на основі якого формувалися відповіді моделі GPT для абітурієнтів університету. У результаті дослідження здійснена програмна реалізація системи для пошуку в базах даних.

У якості предметної області досліджень було обрано тему консультування абітурієнтів. Був створений власний датасет, досліджено різні промпти, використана бібліотека LangChain для пошуку найбільш релевантних фрагментів датасету, розроблено ядро нейроконсультанта абітурієнтів університету, дана експертна оцінка якості відповідей.

Оскільки якість датасету суттєво впливає на результат відповідей моделі, на майбутнє необхідно більше уваги приділити дослідженню таких питань:

- на основі проведених опитувань додаванню до датасету;
- недостатньої або відсутньої інформації;
- сумаризації інформації в датасеті за змістом з подальшим розділенням узагальненої інформації на блоки теж за змістом;
- додавання до датасету інформації з відео матеріалу за рахунок;
- перетворення аудіо сигналу з відео в текстовий вигляд.

В цілому робота показала можливість використання мовних моделей GPT для відповідей на основі власного контенту. Застосування такого підходу для вирішення задачі консультування абітурієнтів університету в автоматичному режимі дозволить налагодити кращу проінформованість абітурієнтів та поліпшити розуміння їх потреб.

Оскільки якість датасету суттєво впливає на результат відповідей моделі, на майбутнє необхідно більше уваги приділити дослідженню таких питань:

- на основі проведених опитувань додаванню до датасету недостатньої або відсутньої інформації;
- сумаризації інформації в датасеті за змістом з подальшим розділенням узагальненої інформації на блоки теж за змістом;
- додавання до датасету інформації з відео матеріалу за рахунок перетворення аудіо сигналу з відео в текстових вигляд.

Результати дослідження опробовано у вигляді 3 тез без доповідей під час V Міжнародній науково-практичній конференції «Trends in science regarding the creation of new teaching methods» [16].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Brandl, R. (2023). Retrieved from <https://www.tooltester.com/en/blog/chatgpt-statistics> .
2. Yakovleva, O., & Nikolaieva, K. (2020). Research Of Descriptor Based Image Normalization And Comparative Analysis Of SURF, SIFT, BRISK, ORB, KAZE, AKAZE Descriptors. *Advanced Information Systems*, 4(4), 89-101. doi:10.20998/2522-9052.2020.4.13.
3. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., & Al-Dhaifallah, M. (2021). Methods of classification of images on the basis of the values of statistical distributions for the composition of structural description components. *IEEE Access*, 9, pp. 92964–92973.
4. Yakovleva, O., Kovtunenکو, A., Liubchenko, V., Honcharenko, V., & Kobylin, O. (2023). Face Detection for Video Surveillance-based Security System (COLINS-2023). In *CEUR Workshop Proceedings (Vol. 3403)*. pp. 69–86.
5. A.R. Kovtunenکو, O.V. Yakovleva, V.A. Lyubchenko, & O.V. Yangolenko (2020) Research on the combined use of mathematical morphology and meshed neural networks for solving the problem of recognizing cinnics. *Visnik Natsional'nogo tehničeskogo universtitutu KHPI* (3). pp. 24–31.
6. Yakovleva, O., Kovač, M., Ardasov, V. & Yeremenko, I. (2023). Study on adding functionality to the Zoom online conference system for monitoring the participant activities. *Public Administration and Regional Development*, 19(1), pp. 158–184.
7. Яковлева, О. В., & Нестерова, Є. П. (2009). Порівняльний аналіз методів характеристик Лавса та матриць збігів у завданнях сегментації текстурних зображень.
8. Яковлева, Е. В., & Кускова, И. В. (2006). Дослідження результатів сегментації зображень методом матриць збігів. *Вісник Національного технічного університету «ХПІ»*. №39 – С.164–171.

9. Yakovleva O., Nebeský L., Liakhov P. (2023). Research methods of texture image analysis to solve the texture search problem. Proceedings of the IV International Scientific and Practical Conference. Vienna, Austria. 2023. pp. 252–261.

10. OpenAI platform. OpenAI Platform. (2023). Retrieved from <https://platform.openai.com>.

11. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). Attention is all you need. arXiv.org. <https://arxiv.org/abs/1706.03762>.

12. 122-Computer science, Educational and professional program "Informatics", Department of Informatics, Faculty of Information and Analytical Technologies and Management, Kharkiv National University of Radio Electronics (2023). Retrieved from <https://informatics.nure.ua/>.

13. LangChain (2023). https://python.langchain.com/docs/get_started.

14. Google. (2023). Dataset_Informatics_Department. Google Docs. https://docs.google.com/document/d/1gN9uhqT41yFjJJ__Q9GOkHwBOuuIBl5EpqMJq6TrjGU/edit#heading=h.bvl9bh32que3.

15. Google. (n.d.-b). Промпт_Informatics_Department. Google Docs. <https://docs.google.com/document/d/1q-pnqGyiY2aJGLPwYDznPz4UEhbDkvtkHLXT7NClom4/edit?usp=sharing>.

16. Yakovleva O., Nebeský L., Kirichenko A. (2023). Using the GPT models for responses based on custom content to develop neural consultant for university applicants. Abstracts of V International Scientific and Practical Conference. Madrid, Spain. pp. 172–178.