Міністерство освіти і науки України

Національний університет «Києво-Могилянська академія»

Факультет інформатики

Кафедра інформатики

Кваліфікаційна робота

освітній ступінь – бакалавр

на тему: «**РОЗРОБКА СИСТЕМИ ДЛЯ ВИЗНАЧЕННЯ СОЦІАЛЬНИХ ТА ЕКОНОМІЧНИХ ТЕНДЕНЦІЙ ШЛЯХОМ АНАЛІЗУ ПЛАТФОРМ З ВІДКРИТИМИ ДАНИМИ**»

Виконав: студент 4-го року навчання,
Освітньої програми «Комп'ютерні науки», 122

Жуковський Ігор Андрійович

Керівник Ющенко Ю.О._____
кандидат фіз.-мат. наук, доцент

Рецензент_R.C. McGeer_____
(прізвище та ініціали)

Кваліфікаційна робота захищена
з оцінкою _____

Секретар ЕК _____
«____» _____ 20____ р.

Київ – 2023

Ministry of Education and Science of Ukraine

NATIONAL UNIVERSITY OF KYIV-MOHYLA ACADEMY

Department of Informatics

Qualification Work

Educational degree - Bachelor's

on topic: «**DEVELOPMENT OF A SYSTEM FOR DETERMINING SOCIAL AND ECONOMIC TRENDS THROUGH ANALYSIS OF OPEN DATA PLATFORMS**»

Performed by: 4th-year student,

Educational Program "Computer Science," 122

Zhukovskyi Ihor Andriiovych

Supervisor: Yushchenko Y.O.

Candidate of Physical and Mathematical Sciences, Associate Professor

Reviewer: R.C. McGeer_____

Qualification work defended

with the grade _____

Secretary of the EC_____

«____» _____ 20____ p.

Kyiv – 2023

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ
Доцент
_____ Ющенко Ю.О.
„__” _____ 2022 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на кваліфікаційну роботу
студенту Жуковському Ігорю Андрійовичу
факультету інформатики 4 курсу бакалаврської програми

ТЕМА: **РОЗРОБКА СИСТЕМИ ДЛЯ ВИЗНАЧЕННЯ СОЦІАЛЬНИХ ТА ЕКОНОМІЧНИХ ТЕНДЕНЦІЙ ШЛЯХОМ АНАЛІЗУ ПЛАТФОРМ З ВІДКРИТИМИ ДАНИМИ**

Зміст ТЧ до кваліфікаційної роботи:
Індивідуальне завдання
Вступ
Розділ 1. Моделі NLP та методи IR для аналізу великої колекції текстових документів
Розділ 2. Сутність запропонованого методу покращення результатів аналізу LLM за допомогою ІЧ-контекстуалізації.
Розділ 3. Реалізація гібридного IR та LLM фреймворку для обробки текстової інформації з використанням набору даних спільноти та документації компанії Anaconda
Розділ 4. Аналіз інформаційних трендів у Twitter від впливових медіаособистостей з використанням розробленого гібридного IR та LLM фреймворку для обробки текстової інформації
Розділ 5. Подальший розвиток проекту.
Висновки
Список літератури

Дата видачі „___” _____ 2022 р.

Керівник_____
(підпис)
Завдання отримав__
(підпис)

Календарний план виконання роботи

№	Назва етапу кваліфікаційної роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на кваліфікаційну роботу	01.10.2022	
2.	Огляд літератури за темою роботи	01.11.2022	
3.	Проведення дослідження	01.12.2022	
4.	Написання програмного застосунку	01.01.2023	
5.	Написання текстової частини	04.04.2023	
6.	Захист кваліфікаційної роботи	30.05.2023	

Студент _____
Керівник _____ “_____”_____ 2022

CONTENT

Анотація

Ця робота присвячена вивченню та інтеграції інформаційно-пошукових (IR) і великих мовних моделей (LLM) для ефективного аналізу та розуміння великомасштабних колекцій текстових документів. У проекті реалізовано інноваційний метод, який поєднує потужні можливості LLM з обробки тексту та точність IR для підвищення ефективності генерації відповідей LLM та точності аналізу текстових даних за рахунок створення контексту для мовної моделі.

Наукова робота представляє новий метод для змішування методів IR з LLM для генерації та контекстуалізації інформації з великих текстових корпусів. Для розробки системи використовувався Python і комбінація бібліотек, таких як PyTorch для LLM і NLTK для IR. Систему було перевірено на різних наборах даних, у тому числі Anaconda Community та Documentation Dataset і даних Twitter від Cryptocurrency Influencers.

Основні розділи роботи описують теоретичні та практичні аспекти як LLM, так і IR, пояснюючи їхні переваги, проблеми та актуальність у галузі науки про дані. Пропонований гібридний метод LLM та IR викладено та детально пояснено разом із його реалізацією.

Завершальна частина дослідження зосереджена на використанні розробленої гібридної системи IR та LLM для аналізу інформаційних трендів у Twitter від Cryptocurrency Influencers. Він забезпечує широке обговорення збору, обробки та аналізу даних за допомогою реалізованої структури. Робота завершується дорожньою картою подальшого розвитку та застосування розробленого методу.

Письмова частина дипломної роботи містить огляд можливостей і обмежень існуючих інструментів НЛП для різних завдань аналізу тексту, а потім заглиблюється в суть запропонованого методу. Він також дає огляд реалізації гібридної структури IR та LLM за допомогою набору даних спільноти та документації Anaconda. Письмова частина дисертації детально аналізує інформаційні тренди в Twitter від Cryptocurrency Influencers з використанням розробленого фреймворку IR та LLM.

Ключові слова: великі мовні моделі, інформаційний пошук, аналіз текстових документів, аналіз даних у Twitter, суб’єкти впливу на криптовалюту, обробка природної мови, наука про дані, Python.

Anotation

This work is dedicated to the exploration and integration of Information Retrieval (IR) and Large Language Models (LLMs) for effective analysis and understanding of large-scale text document collections. The project implements an innovative method that unites the powerful text processing capabilities of LLMs and the precision of IR to improve the efficiency and accuracy of text data analysis tasks.

The thesis introduces a novel framework for blending IR techniques with LLMs to generate and contextualize information from large text corpora. Python was used for the development of the system and a combination of libraries such as PyTorch for LLMs and NLTK for IR. The system has been tested on various datasets, including Anaconda Community and Documentation Dataset and Twitter data from Cryptocurrency Influencers.

The main sections of the work describe the theoretical and practical aspects of both LLMs and IR, explaining their advantages, challenges, and relevance in the data science field. The proposed hybrid method of LLM and IR is outlined and explained in detail, along with its implementation.

The final part of the research focuses on the use of the developed hybrid IR and LLM framework for the analysis of information trends in Twitter from Cryptocurrency Influencers. It provides an extensive discussion on data collection, processing, and analysis using the implemented framework. The work concludes with a roadmap for further development and application of the developed method.

The written part of the thesis provides an overview of the capabilities and limitations of existing NLP tools for various text analysis tasks and then delves into the essence of the proposed method. It also gives an overview of the implementation of the hybrid IR and LLM framework using the Anaconda Community and Documentation Dataset. The written part of the thesis further details the analysis of informational trends in Twitter from Cryptocurrency Influencers using the developed IR and LLM framework.

Keywords: Large Language Models, Information Retrieval, Text Document Analysis, Twitter Data Analysis, Cryptocurrency Influencers, Natural Language Processing, Data Science, Python.

ВСТУП

Швидкий розвиток цифрових технологій перетворив аналіз соціальних та економічних тенденцій у все складніше завдання. З'явлення проривів у природній обробці мови (Natural Language Processing, NLP), таких як моделі GPT та інші моделі LLM, значно спростило розробку рішень NLP для аналізу тенденцій.

Проте моделі GPT також принесли нову проблему: неможливість обробки великих текстів у одному запиті через обмеження на розмір вводу. Щоб вирішити цю проблему, було розроблено новий підхід, який поєднує класичні методи інформаційного пошуку (Information Retrieval, IR) з останніми моделями GPT. Шляхом визначення найбільш відповідних фрагментів текстових даних з колекції документів, що стосуються певного запиту, система може створити навчальну вибірку потрібного розміру та передати її моделям GPT для виконання завдань NLP.

Ця дипломна робота презентує систему, яка ефективно виявляє та аналізує тенденції у текстових даних з різних джерел відкритих даних.

Актуальність теми: Нагальна потреба у поєднанні класичних методів інформаційного пошуку та найновіших моделей NLP має потенціал революціонізувати галузь, що робить цю дипломну роботу високоактуальною.

Мета дослідження: Розробити та реалізувати хмарну систему пошуку та аналітики, яка використовує методи інформаційного пошуку для покращення продуктивності та якості результатів генерації відповідей моделей LLM.

Для досягнення цієї мети **були визначені наступні наукові задачі:**

1. Дослідити обмеження поточних моделей NLP, таких як родина GPT та інші моделі LLM, у обробці великих текстів. Проаналізувати існуючі методи та підходи інформаційного пошуку для пошуку, витягування, обробки та аналізу текстових даних у великій колекції документів, що стосуються запиту.
2. Створити хмарну систему пошуку та аналітики, яка поєднує класичні методи інформаційного пошуку з сучасними моделями NLP для ефективного аналізу великих корпусів тексту та надання відповідей у природній мові.
3. Реалізувати та протестувати розроблену систему, оцінюючи її продуктивність та точність у виявленні тенденцій і закономірностей, а також забезпечуючи можливості інформаційного пошуку.
4. Порівняти продуктивність системи з існуючими підходами та зробити висновки про її застосовність та ефективність.

Об'єкт дослідження: Знайти спосіб покращити результати генерації моделей LLM для конкретних тем шляхом створення контексту на основі інформації та запитів, наданих користувачами, з метою ефективного виявлення та аналізу соціальних та економічних тенденцій у корпусах текстового контенту.

Методи дослідження: Для досягнення поставлених наукових задач були використані наступні методи: огляд літератури, розробка алгоритмів, реалізація програмного забезпечення, аналіз даних, порівняльний аналіз, експериментальна перевірка та консультації з професором із комп'ютерних наук Університету Вікторії.

Наукова новизна отриманих результатів: У дипломній роботі презентовано новий підхід, який поєднує класичні методи інформаційного пошуку з передовими моделями NLP, ефективно вирішуючи обмеження на розмір вводу і дозволяючи виявляти та аналізувати тенденції у великих корпусах тексту.

Апробація результатів: Наукова робота була високо оцінена науковим керівником кандидатом фіз.-мат. Наук та доцентом Ющенко Юрієм Олексійович і отримала позитивні відгуки від Adjunct professor Університету Вікторії, доктора філософії з електротехніки та комп'ютерних наук Університету Каліфорнії в Берклі, ветерана DARPA Річарда Чарльза МакГіра.

Практичне значення отриманих результатів: Деякі продукти будуть розроблені на основі наукових висновків цієї роботи у співпраці з компанією Galyleo (engageLivley), що є C-Corp зареєстрованою в Делавері з представниками в районі затоки Сан-Франциско, Каліфорнія, США.

INTRODUCTION

The rapid advancement of digital technology has transformed the analysis of social and economic trends into an increasingly intricate task. The advent of breakthroughs in natural language processing (NLP), such as GPT and other LLM models, has significantly simplified the development of NLP solutions for trend analysis.

However, the GPT models have also introduced a new problem: the inability to process large texts in a single query due to input size restrictions. To address this issue, a novel approach has been developed that combines classic information retrieval (IR) techniques with the latest GPT models. By identifying the most relevant text data pieces from a document collection related to a specific query, the system can create a training data frame of the appropriate size and pass it to the GPT models to execute the NLP tasks.

This diploma demonstrates a system that effectively identifies and analyzes trends in text data from diverse open data sources.

Relevance of the topic: The urgent need for the fusion of classic IR techniques and the most recent NLP models has the potential to revolutionize the field, making this diploma work highly relevant.

The purpose of the research: To develop and implement a cloud-based searching and analytics system that utilizes information retrieval techniques to improve the performance and quality of answer generation results from LLM models.

To achieve this goal, the following **scientific tasks have been determined:**

1. Investigate the limitations of current NLP models, such as the GPT family and other LLMs, in processing large texts. Analyze existing IR methods and

approaches to search, extract, process, and analyze text data in large collections of documents relevant to the query.

2. Create a cloud-based searching and analytics system that combines classic IR techniques with modern NLP models to effectively analyze large text corpora and provide answers in natural language.
3. Implement and test the developed system, assessing its performance and accuracy in identifying trends and patterns, as well as providing IR search capabilities.
4. Compare the performance of the system with existing approaches and draw conclusions about its applicability and effectiveness.

The object of study: To find a way to improve the generation results of LLMs for specific topics by creating context based on information and queries provided by humans, in order to efficiently identify and analyze social and economic trends in large text content datasets.

Research methods: To achieve the research tasks, the following methods were used: literature review, algorithm development, software implementation, data analysis, comparative analysis, experimental validation, and consultations with a CS Professor from the University of Victoria.

The scientific novelty of the obtained results: The diploma work presents a novel approach that merges classic IR techniques with cutting-edge NLP models, effectively addressing the input size limitations and enabling the identification and analysis of trends in large text corpora.

Approbation of the results: The scientific work was highly appreciated by the scientific supervisor, and also received favorable feedback from Adjunct ProfessorAdjunct Professor University of Victoria, University of California, Berkeley PhD, Electrical Engineering and Computer Science, DARPA Veteran.

Practical significance of the obtained results: Some products will be developed based on the scientific findings of this work in collaboration with Galyleo (engageLivley), a C-Corp Delaware company with representatives in the San Francisco Bay Area, California, USA. The actual product releases are covered by a non-disclosure agreement (NDA). Currently, Galyleo hosts and manages the actual products as well as the testing environment for the diploma.

LIST OF ACCEPTED ABBREVIATIONS

DL - Deep Learning;

NLP - Natural Language Processing;

GPT - Generative Pre-trained Transformer;

LLM - Large Language Model;

IR - Information Retrieval;

R&D - Research and Development;

CS - Cosin Similarity;

Da Vinci – LLM created by OpenAI;

ADA - Autoregressive Distribution Alignment;

TF-IDF - Term Frequency-Inverse Document Frequency.

CHAPTER 1: NLP models and IR methods for analysis of large text document collection

1.1 The relevance of analyzing large text corpora in data science tasks

In the modern IT world, information permeates every aspect of our lives, spanning thousands of petabytes and reaching billions of individuals worldwide at an astonishing speed. With such an abundance of data, the proper extraction, processing, and analysis of information have become fundamental in building mathematical models with the highest accuracy for predictions. It is in this context that data scientists emerge as the oracles of our time, leveraging their expertise to navigate the vast sea of data and derive valuable insights.

So the main scientific aim of this work is to define the best way to data trends analysis in the massive volume of text data.

Project Cassandra [1] stands as a compelling example supporting this notion. By implementing advanced information extraction, processing, and analysis techniques, the project succeeds in predicting potential conflicts and societal unrest by examining literature texts. Through the careful scoring of books and assessment of various indicators, Project Cassandra sought to uncover patterns and insights that could shed light on future geopolitical developments. This endeavor showcases the pivotal role of data scientists in harnessing the power of information to make informed predictions and decisions.

The partial success of the project demonstrates that in the era of information overload, effective information extraction, processing, and analysis are paramount. Data scientists possess the necessary skills to transform this wealth of data into meaningful and actionable insights.

However, as is often the case in significant data science tasks, a major challenge arises when it comes to translating textual data into a format that is suitable for computer processing. This step is crucial to leverage the full potential of computational algorithms and techniques in analyzing and extracting insights from large text collections. In this task, the integration of Natural Language Processing (NLP) and Information Retrieval (IR) techniques plays a vital role in effectively transforming and analyzing text data for data science purposes.

1.2 Analysis the possibilities and limitations of existing NLP tools for various text analysis.

Text normalization is an essential and integral part of Natural Language Processing (NLP) that plays a crucial role in converting text into a format that machines can readily understand and process. It is a fundamental step in NLP pipelines, enabling effective comprehension and analysis of text by computational algorithms. By normalizing text, removing inconsistencies, and representing it in a standardized manner, machines can efficiently handle and interpret the information contained within the text.

Text normalization, a fundamental task in Natural Language Processing (NLP), encompasses a variety of established techniques like stemming, lemmatization, TF-IDF (Term Frequency-Inverse Document Frequency), and more. These methods are employed to convert raw textual data into a format that can be effectively comprehended and analyzed by computer algorithms.

Stemming reduces words to their root form, disregarding grammatical variations. Lemmatization, on the other hand, aims to derive the base or dictionary form of a word, considering its part of speech. TF-IDF is a statistical measure used to evaluate the importance of a term within a document or a corpus by considering its frequency and inverse document frequency.

Table 1. Summary of NLP Techniques

Method	Brief Description	Pros	Con
Stemming	Reduces words to their root form, ignoring grammatical variations.	- Simplicity and speed of implementation.	- Loss of precision due to overgeneralization.
Lemmatization	Derives the base or dictionary form of a word, considering its part of speech.	- Produces linguistically valid word forms.	- More computationally intensive compared to stemming.
TF-IDF	Measures the importance of a term in a document or corpus based on inversed term frequency	- Provides a statistical measure of term relevance.	- Limited to term frequency, may not capture semantic relationships.

Despite the benefits of these techniques, their implementation can be time-consuming, leading to a significant expenditure of developer resources. Developers worldwide spend countless hours each day on this task alone, emphasizing the need for more efficient solutions.

The introduction of large language models (LLMs) like the Generative Pretrained Transformer (GPT) series by OpenAI has revolutionized the NLP landscape. These models are trained on vast corpora of text data, incorporating expansive and diverse lexical knowledge from various domains and industries. As a result, they come with precompiled thesauruses, eliminating the need for developers to manually compile this information. Additionally, these models offer the advantage of accessibility through APIs, which means developers do not need to deploy their own language models. Instead, they can leverage the pre-trained models via API calls, thereby simplifying the process and saving significant time and resources.

However, despite the remarkable capabilities of these LLMs, they come with their set of constraints. One significant limitation is the restriction on the maximum number of tokens that can be processed in each API request. This constraint is a byproduct of the need to manage high computational costs and to serve a large user infrastructure efficiently. As a result, processing large volumes of text for analysis using only GPT models is not feasible.

Table 2. Various GPT models restrictions

Model	Maximum Token Limit
GPT-2 (345M parameters)	1,536
GPT-3 (175B parameters)	3,072
GPT-3.5 (175B parameters)	4,096
GPT-4 (??B parameters)	8,192
Other LLMs (various parameters)	Varies

These token limits represent the maximum number of tokens that can be processed in a single API call. If the input text exceeds the token limit, it needs to be divided into smaller segments and processed separately. Consequently, handling long contexts or documents with LLMs becomes a more complex task.

To overcome these limitations, one approach is to employ Information Retrieval (IR) techniques. IR methods involve performing an initial search based on the user's query to identify relevant portions of text from a larger document collection. By extracting

and focusing on the most important segments, the context can be narrowed down to fit within the LLM's token limit.

Once the context has been extracted, it can be processed by the LLM to provide accurate and comprehensive analysis. This combination of IR and LLM techniques enables efficient and effective text analysis, even for large-scale tasks.

It is worth noting that the token limits mentioned in the table may change as newer iterations of LLMs are developed. Each new version may introduce higher token limits, allowing for the processing of longer contexts. Therefore, it is essential to refer to the documentation or official resources of the specific LLM being used to obtain the most up-to-date information on token limits and any associated guidelines.

In response to this challenge, Information Retrieval (IR) methods present a viable solution. By using IR techniques, we can execute a preliminary search based on the user's query to identify and extract the most relevant chunks of text from a large document collection. This narrowed down context can then be processed by the LLM, ensuring that the input remains within the model's token limit while still providing accurate and comprehensive analysis. This approach combines the efficiency of IR methods with the linguistic prowess of LLMs, promising a powerful solution for large-scale text analysis tasks.

Chapter 1.

Chapter 1 introduces the relevance of analyzing large text corpora in data science tasks and highlights the importance of effective information extraction, processing, and analysis. It discusses the challenges of translating textual data into a format suitable for computer processing, emphasizing the integration of Natural Language Processing (NLP) and Information Retrieval (IR) techniques as vital for transforming and analyzing text data.

The chapter explores various text normalization techniques used in NLP, such as stemming, lemmatization, and TF-IDF, which convert raw textual data into a standardized format. While these techniques offer benefits, their implementation can be time-consuming and resource-intensive, creating a need for more efficient solutions.

The introduction of large language models (LLMs) like the GPT series by OpenAI revolutionized NLP by offering pre-trained models with extensive lexical knowledge. These models eliminate the manual compilation of thesauruses and provide accessibility through APIs, simplifying the development process and saving time and resources.

However, LLMs have constraints, notably the restriction on the maximum number of tokens that can be processed in each API request. To address this limitation, the chapter proposes the use of IR techniques, which involve performing an initial search to identify relevant portions of text and extract them for processing within the LLM's token limit. This combination of IR and LLM techniques enables efficient and effective text analysis, even for large-scale tasks.

The chapter concludes by highlighting the importance of referring to the specific LLM's documentation or official resources for the most up-to-date information on token limits and guidelines.

With the understanding of text normalization and the constraints of LLMs established in Chapter 1, the subsequent chapter will delve into a solution chapter that addresses these limitations and presents efficient approaches to extend mentioned limitations.

CHAPTER 2: The essence of the proposed method to improve LLMs analysis results using IR contextualization.

2.1 Introduction

In the realm of computational linguistics and data science, the quest for more effective methods of text analysis is an ongoing endeavor. This pursuit has led to the development and refinement of various scientific approaches and methodologies, each with its unique strengths and limitations. Central to the discourse of this diploma thesis is the exploration and integration of two such powerful methodologies: Large Language Models (LLMs) and Information Retrieval (IR) techniques.

Large Language Models, particularly those based on the transformer architecture like the GPT series developed by OpenAI, have displayed remarkable capabilities in understanding and generating human-like text. When juxtaposed with traditional IR techniques such as Vector Space Models (VSMs), these LLMs can be deployed in unique and innovative ways to improve the efficiency and accuracy of text analysis.

This chapter will elucidate the theoretical underpinnings of these methodologies and provide an in-depth exploration of their integration in the proposed method. The ensuing sections will also illustrate how this integrated approach can be used to enhance the performance of LLMs by providing more contextually relevant information, leading to improved text analysis results.

2.2 Large Language Models: An In-Depth Overview

2.2.1 Evolution of Large Language Models

The inception of Large Language Models (LLMs) dates back to the early 2000s when machine learning began being applied to Natural Language Processing (NLP). The field started with relatively simple models such as Naive Bayes and Support Vector Machines for text classification tasks. However, with the advent of more advanced

machine learning techniques, including deep learning, the complexity and effectiveness of language models have significantly improved [***2***].
The milestone moment for LLMs was arguably the development of the transformer architecture, which powered models such as GPT-3 and GPT-4. Transformers replaced the sequential processing inherent in previous architectures, such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, with a parallelized, attention-based approach. The transformative effect of this architecture led to the development of increasingly large and powerful models, marking a new era in the field of NLP [***3***].

2.2.2 Understanding the Transformer Architecture

The transformer architecture is a novel approach in the realm of deep learning, replacing sequential processing with a parallelized, attention-based model. This architecture is fundamentally built around the concept of 'self-attention', or the ability of the model to weigh the importance of different words in a sequence when generating a response. This means the transformer doesn't just consider the current word or a fixed window around it, but can refer back to any part of the input when generating its output[***4***].

This innovation solves two significant challenges with prior architectures. First, it resolves the problem of 'long-term dependencies', where the context of words early in a sequence can be lost by the time the model processes the end of the sequence. Second, it enables much faster and more efficient processing of text data due to the parallel nature of the attention mechanism[***5***].

2.2.3 The Power of Large Language Models

What truly sets LLMs like GPT-3 and GPT-4 apart is not just their innovative architecture, but their size and the extensive training data used to fine-tune them. These models consist of billions, or even trillions, of parameters that are learned from vast amounts of text data sourced from the internet. This wealth of training data

enables the models to learn and mimic complex language patterns, effectively capturing the nuances of human language[***6***].

The extensive training process allows these models to generate coherent and contextually appropriate responses. This is a significant leap from previous models, which often struggled with maintaining context over longer text sequences.

Moreover, their unparalleled size and complexity enable them to perform a range of NLP tasks without needing task-specific training data. Instead, they can generalize from the patterns they learned during training, making them capable of 'few-shot' or even 'zero-shot' learning[***7***].

2.2.4 The Impact on Natural Language Processing

The introduction of LLMs has had a profound impact on the field of NLP. With their unprecedented capabilities, they have been used for various tasks, including machine translation, text summarization, sentiment analysis, and more. Moreover, their ability to understand and generate text that closely resembles human language has opened up new avenues for applications such as chatbots, personal assistants, and content generation.

Furthermore, their potential extends beyond typical NLP tasks. Given their ability to generalize from learned patterns, these models have also been used for tasks like code generation or solving simple math problems, pushing the boundaries of what we typically consider NLP[***8***].

However, it's crucial to note that while LLMs represent a significant step forward in NLP, they are not without their limitations. These models can sometimes generate text that is plausible-sounding but factually incorrect, and they can inadvertently reflect biases present in their training data. Therefore, understanding and addressing these limitations remains an active area of research[***9***].

In summary, the advent of Large Language Models has dramatically transformed the landscape of natural language processing. As our understanding of these models and their potential applications continue to grow, they are poised to play an increasingly important role in many aspects of our digital lives.

2.3 Information Retrieval Techniques: A Deep Dive

2.3.1 The Birth of Information Retrieval

Information Retrieval (IR) is an interdisciplinary field with roots in computer science, information science, and cognitive psychology. The primary focus of IR is to retrieve relevant and useful information from a vast dataset or corpus. The advent of the digital age and the explosion of online data have transformed the field, necessitating increasingly sophisticated methods for retrieving, organizing, and delivering information [10].

Historically, IR began with relatively simple techniques like boolean search and term frequency measures. The rise of the internet, however, saw a rapid evolution of IR, with the need for better techniques to navigate the enormous volume of data available online. This led to the development of more advanced techniques, including ranking algorithms, clustering, indexing, and the Vector Space Model (VSM) [11].

2.3.2 The Vector Space Model: A Pillar of IR

The Vector Space Model is a pivotal concept in modern IR systems. In this model, each document within a corpus is represented as a vector in a multidimensional space, with each unique term in the corpus forming a dimension. The position of a document in this space is determined by the frequency or importance of each term within the document. Essentially, each document has its own unique 'fingerprint' within the vector space, determined by its content [12].

The VSM not only represents the content of documents quantitatively but also allows for effective document comparison. The semantic distance between vectors represents the relative similarity or dissimilarity of the documents they represent. This facilitates the process of identifying and retrieving documents that are contextually relevant to a given query [13].

2.3.3 Cosine Similarity: A Measure of Relevance

To measure the degree of similarity between a query and a document (both represented as vectors in the VSM), IR systems often use a metric known as cosine similarity. Cosine similarity essentially measures the cosine of the angle between two vectors. If the vectors are identical, their cosine similarity is 1 (cosine of 0 degrees), and if they are completely dissimilar, their cosine similarity is 0 (cosine of 90 degrees) [14].

In an IR context, this metric is used to determine how 'similar' two documents are in terms of their content or how relevant a particular document is to a user's query. By comparing the query vector with the document vectors in the corpus, the system can effectively rank documents based on their relevance, thereby enhancing the efficiency and accuracy of the retrieval process [15].

2.3.4 Advanced Techniques and Modern IR

While VSM and cosine similarity are fundamental to IR, the field has evolved to incorporate more complex techniques. Ranking algorithms, such as Google's PageRank, use link analysis to determine the importance of different web pages, improving search results for users. Similarly, techniques like latent semantic indexing (LSI) use singular value decomposition to uncover underlying semantic relationships between words, enhancing the ability of IR systems to understand context and improve the relevance of search results [16].

Moreover, with the advancement in machine learning and NLP, IR systems have begun to incorporate these technologies to further improve their performance. Techniques like word embeddings (e.g., Word2Vec, GloVe) and LLMs are being used to better understand the semantic context of words and documents, significantly enhancing the capability of modern IR systems [17].

As we move forward in the digital age, the importance of effective IR systems will continue to grow. Whether it's finding relevant academic papers, locating a helpful blog post, or retrieving a specific email, IR is a critical component of our digital lives.

2.4 Integrating LLMs and IR: The Proposed Method

2.4.1 The Vision: LLMs and IR Unite

The method proposed in this study envisages a profound integration of Large Language Models (LLMs) and Information Retrieval (IR) techniques to amplify the precision and efficiency of text analysis systems. This innovative approach essentially combines the semantic understanding capabilities of LLMs with the adeptness of IR in retrieving relevant information from a large dataset [18].

2.4.2 Initial Stage: Information Retrieval

The first phase of the proposed methodology involves deploying an IR system to curate relevant documents from a more extensive corpus. This phase banks on the Vector Space Model (VSM) to represent the entire corpus and the cosine similarity metric to single out documents that exhibit the highest similarity to the user's query. In the VSM, every unique term and document correspond to a dimension and vector respectively. When a user's query is represented as a vector, the cosine similarity between the query vector and document vectors can be calculated, signifying the relevance of the documents to the query [19].

2.4.3 Transition: Feeding LLMs with Retrieved Information

The ensuing stage incorporates Large Language Models into the process. The documents or chunks of text retrieved by the IR system are fed into the LLM as inputs. Given the contextual understanding capabilities of LLMs, they can process the

inputs to generate text that aligns closely with the content and context of the input text [20].

2.4.4 Advantages of the Proposed Method

The amalgamation of LLMs and IR techniques in the proposed methodology offers several potential benefits. Firstly, it allows for more effective management of token constraints inherent to LLMs. As LLMs can handle only a limited number of tokens at once, focusing on the most relevant parts of the corpus can enhance their efficacy and efficiency [21].

Secondly, this approach can potentially augment the precision and relevance of the responses generated by the LLM. By providing the model with a more focused subset of the corpus that aligns closely with the user's query, the generated responses are likely to be more contextually appropriate and accurate [22].

Thirdly, the proposed system is inherently adaptable. It can adjust to different domains as the corpus can be easily replaced or augmented with new data. This flexibility makes the proposed methodology broadly applicable, making it a promising direction for future text analysis systems [23].

2.4.5 Looking Forward: The Potential of Integrated LLMs and IR Systems

As we move into an era of increasing digitalization, where the volume of text data is growing exponentially, the integration of LLMs and IR could prove to be a significant breakthrough. By combining the power of LLMs in understanding complex language patterns with the prowess of IR in retrieving relevant documents, we can hope to develop highly effective and accurate text analysis systems that can handle the vast amount of available data and cater to a diverse range of user queries.

In the subsequent chapters, we will delve into the technical aspects of the proposed methodology, providing a detailed analysis of how LLMs and IR techniques can be combined. We will also present a comprehensive evaluation of the proposed methodology, including its performance, strengths, and potential areas for improvement.

CHAPTER 2: Conclusion

In this chapter, we delved into the essence of the proposed method to improve Large Language Models (LLMs) analysis results using Information Retrieval (IR) contextualization. We explored the evolution of LLMs, understanding the transformative impact of the transformer architecture on Natural Language Processing (NLP). Additionally, we discussed the birth of Information Retrieval, highlighting the Vector Space Model and cosine similarity as fundamental techniques in IR.

The integration of LLMs and IR offers a promising approach to enhance text analysis systems. By combining the semantic understanding capabilities of LLMs with the adeptness of IR in retrieving relevant information, we can achieve improved precision and efficiency in analysis results. The proposed method involves the initial stage of information retrieval, followed by feeding the retrieved information to LLMs for further processing and generating contextually appropriate responses.

The advantages of this integrated approach are manifold. It allows for effective management of token constraints in LLMs, enhances the precision and relevance of responses generated by LLMs, and offers adaptability across different domains. Integrated LLMs and IR systems have the potential to revolutionize text analysis, opening new avenues for practical applications and advancements in the field.

Looking forward, the integration of LLMs and IR holds great promise. Further research and development in this area can lead to the creation of more powerful and intelligent systems capable of understanding and generating human-like text. The potential impact of integrated LLMs and IR systems extends beyond NLP, with implications for fields such as code generation and problem-solving.

In the next chapters, we will delve into the practical implementation of the proposed method, including the data collection process, data processing for IR, information retrieval, and the utilization of LLMs for question answering. Through detailed explanations and examples, we will showcase the effectiveness and potential of this integrated approach.

CHAPTER 3: Implementation of the Hybrid IR and LLM Framework Using Anaconda Community and Documentation Dataset

This chapter outlines the process of realizing a framework that seamlessly combines Information Retrieval (IR) and Large Language Models (LLMs), specifically the Da Vinci model. For demonstration, were used articles from the Anaconda Community and Documentation as our document dataset.

3.1 Data Collection

3.1.1 Objective

In the context of the hybrid IR and LLM system, our objective in this stage is to gather as much pertinent data as possible to solidify the effectiveness of the IR process. The data that we seek to collect consists of articles from the Anaconda Community and Documentation [24]. These articles serve as the database for the IR system and will subsequently be processed and used for answering queries.

3.1.2 Web Crawling Methodology

To procure the data efficiently, we employ a technique known as web crawling. Web crawling is a method used to extract large amounts of data from websites quickly and accurately.

We design a web crawler specifically for this task. The crawler is programmed to navigate through the Anaconda Community domain and extract the text content of the articles. This focused approach ensures that only relevant articles within the domain are included, thereby ensuring the specificity and relevance of our data collection.

The following code block demonstrates a simple web crawler using the BeautifulSoup [25] and requests [26] libraries in Python:

```
import requests
from bs4 import BeautifulSoup
import time

def crawl_page(url):
    # send a GET request to the URL
    response = requests.get(url)
    # if the GET request is successful, the status code will be 200
    if response.status_code == 200:
        # get the content of the response
        page_content = response.content
        # create a BeautifulSoup object and specify the parser
        soup = BeautifulSoup(page_content, 'html.parser')
        # find the content of the article
```

```
        article_content = soup.find('div', {'class': 'article-content'})
        return article_content.text
    else:
        return None

# Define the URLs of the articles to be scraped
urls = ['https://www.anaconda.com/article-url1', 'https://www.anaconda.com/article-url2']

# List to hold the content of articles
articles = []

# Crawl each URL and get the article content
for url in urls:
    article = crawl_page(url)
    if article:
        articles.append(article)
    # Sleep to prevent overloading the server
    time.sleep(1)
```

Code Example 1. Data crawler from open source

3.2 Data Processing for Information Retrieval

3.2.1 Overview

The second phase in building our hybrid framework is pre-processing the data for the Information Retrieval (IR) process. The purpose of this stage is to convert raw data into a suitable form that is efficient and effective for the subsequent stages of the IR

system. As raw data can be unstructured and noisy, processing it becomes a prerequisite to achieve precise retrieval results.

3.2.2 Data Storage

The collected data, which consists of text from articles, is stored in individual text files. Each file is named and saved according to the URL of the respective article. This scheme of organization allows for easy reference and retrieval of data.

3.2.3 Tokenization and Indexing

The next crucial step involves tokenizing and indexing the data. Tokenization is a fundamental task in natural language processing that involves breaking down text into units called tokens. Tokens can be words, phrases, symbols, or other significant elements that aid in understanding the context of the text. For example, a sentence such as "Natural language processing is interesting" would be tokenized into ["Natural", "language", "processing", "is", "interesting"].

The tokenization process uses the tiktoken library [27], which is designed to work effectively with the ada-002 model. Tiktoken handles tokenization of the text data efficiently and ensures compatibility with the tokenization process of the ada-002 model.

Here is an example of how you can use tiktoken to tokenize a text:

```
from tiktoken import Tokenizer
tokenizer = Tokenizer()

text = "Natural language processing is interesting."
tokens = tokenizer.tokenize(text)
print(tokens)
```

Code Example 2. Tokenizer Example

Once tokenized, the textual data are then segmented into manageable chunks using the **split_into_many_mod** function from the tiktoken library. The function ensures that no single chunk of text exceeds a specified maximum token limit. This helps keep the token count within the manageable range for the LLM, thereby increasing the efficiency of the overall system.

```
from tiktoken import Tokenizer

def split_into_many_mod(text, max_tokens):
    tokens = tokenizer.tokenize(text)
    for i in range(0, len(tokens), max_tokens):
        yield tokens[i:i + max_tokens]

tokenizer = Tokenizer()
max_tokens = 100  # assuming a max token limit
text = "..."  # input text to be tokenized and split
chunks = list(split_into_many_mod(text, max_tokens))
```

Code Example 3. Split_into_many_mod Example

3.3 Information Retrieval

3.3.1 Overview

Once our data is appropriately tokenized and processed, the next phase of our hybrid framework is Information Retrieval (IR). This process helps us extract and identify the most relevant documents from our data corpus, given a specific user query. The underlying principle for this IR process is the use of word and document embeddings.

3.3.2 Embeddings

Embeddings are a multidimensional representation of words or documents that capture the semantic and syntactic relationships between them [28]. They allow words or documents with similar meanings to be represented in similar ways, preserving the contextual relationships within the text data. This makes embeddings particularly suitable for information retrieval tasks, as they allow for context-aware matching between queries and documents.

In our case, we use the 'text-embedding-ada-002' engine to generate embeddings of our text data [29]. This engine is a pre-trained model that takes in text and outputs a numeric vector that represents the text. For each document in our dataset, we generate its embeddings and store them for future use during the retrieval process.

Here's a simplified example of how we might generate embeddings for a text using a hypothetical 'text-embedding' engine:

```
for ind, x in enumerate(list(df["text"])):
    if ind % 50 == 0:
        # sleep is required to avoid cases of API overload
        time.sleep(2)
    embeddings_lst.append(openai.Embedding.create(input=x, engine='text-
embedding-ada-002')['data'][0]['embedding'])


df["embeddings"] = pd.Series(embeddings_lst)

df.to_csv('processed/' + domain + '_embeddings.csv')
```

Code Example 4. Embeddings creation Example

3.3.3 Query Processing and Document Retrieval

When a user submits a query, the same 'text-embedding-ada-002' engine is used to generate embeddings for the query. We then find the most similar document(s) to the query by comparing the query embeddings with the document embeddings. This is achieved using a measure known as cosine similarity [30]. Cosine similarity measures the cosine of the angle between two vectors. In our case, these vectors are the embeddings of the query and the documents. Documents with a high cosine similarity are considered relevant to the query.

To illustrate, consider the following example where we compute the cosine similarity between the query and document embeddings:

```
from sklearn.metrics.pairwise import cosine_similarity

# Given embeddings of the query and documents
query_embedding = embedding_engine.get_embedding("What is Anaconda?")
document_embeddings = [embedding_engine.get_embedding(doc) for doc in
documents]

# Compute cosine similarity
similarities = cosine_similarity([query_embedding], document_embeddings)

# Find the document(s) with the highest similarity
most_similar_document_index = similarities.argmax()
most_similar_document = documents[most_similar_document_index]

# Output the most similar document
```

```
print("Most similar document to the query:", most_similar_document)
```

Code Example 5. Cosine Similarity Example

In the end, this process produces a pool of relevant documents. These documents are then sent to the LLM for further processing and generation of a response to the user query.

3.4 Context Generation and Question Answering using LLM

3.4.1 Overview

The final step of our hybrid framework is to leverage a Large Language Model (LLM) – specifically the Da Vinci model – to generate a response to the user query. The foundation of this step is the context generated from the relevant documents retrieved during the IR process.

3.4.2 Context Generation

The documents identified in the IR stage are more than just a list of potential answers; they provide a context that helps the LLM understand the user's query more thoroughly. This context is essentially a coherent combination of the retrieved documents and encapsulates the information necessary to answer the query.

We could create the context as follows:

```
def create_context(
    question, df, max_len=2000
):
    """
```

```
    Create a context for a question by finding the most similar context from the dataframe
    """

    # Get the embeddings for the question
    q_embeddings = openai.Embedding.create(input=question, engine='text-embedding-ada-002')['data'][0]['embedding']

    # Get the distances from the embeddings
    df['distances'] = distances_from_embeddings(q_embeddings, df['embeddings'].values, distance_metric='cosine')

    returns = []
    reference_links = []
    cur_len = 0

    # Sort by distance and add the text to the context until the context is too long
    for i, row in df.sort_values('distances', ascending=True).iterrows():

        # Add the length of the text to the current length
        cur_len += row['n_tokens'] + 4

        # If the context is too long, break
        if cur_len > max_len:
            break

        # Else add it to the text that is being returned
        returns.append(row["text"])
```

```
        #reference_links.append(fname_to_link_mapping['text/' + domain + "/" +
row["title"].strip()])
        reference_links.append(fname_to_link_mapping[row["title"].strip()])


    # Return the context
    return {"context_text": "\n\n###\n\n".join(returns), "reference_links":
list(set(reference_links))}
```

Code Example 6. Context Creation Example

3.4.3 Question Answering using LLM

Once the context is prepared, it's fed into the LLM along with the user query. The Da Vinci model, designed and trained by OpenAI, generates a text-based answer using the input query and context. The model is primed to play the role of a customer support specialist, ensuring the responses it generates are not only accurate but also helpful and actionable.

Here's an example of how we might use the Da Vinci model to generate an answer:

```
def answer_question(
    df,
    model="text-davinci-003",
    question="Am I allowed to publish model outputs to Twitter, without a human
review?",
    max_len=2000,
    debug=False,
    max_tokens=150,
    stop_sequence=None,
    human_support_link=None
```

```
):
    """
    Answer a question based on the most similar context from the dataframe texts
    """
    context_results = create_context(
        question,
        df,
        max_len=max_len
    )

    context = context_results["context_text"]
    reference_links = context_results["reference_links"]

    # If debug, print the raw model response
    if debug:
        print("Context:\n" + context)
        print("\n\n")

    try:

        response = openai.ChatCompletion.create(
            messages=[
                {"role": "user", "content": f"Act as a customer support specialist. Give a helpful actionable answer. Answer the question based on the context below, and if the question can't be answered based on the context, say \"I am not sure, but I will attach links that might be hepful.\"\n\nContext: {context}\n\n---\n\nQuestion: {question}\nAnswer:"}
            ],
                    temperature=0,
                    max_tokens=800,
```

```
            top_p=1,
            frequency_penalty=0,
            presence_penalty=0,
            stop=None,
            model='gpt-3.5-turbo-0301',
        )

    text_answer = response["choices"][0]["message"]['content'].strip()

    return {"text_answer": text_answer, "reference_links": reference_links,
"human_support_link": human_support_link}
```

Code Example 7. How to answer questions using context creation

CHAPTER 3: Conclusion

In this chapter, we focused on the implementation of the hybrid Information Retrieval (IR) and Large Language Model (LLM) framework using the Anaconda Community and Documentation dataset. We discussed the data collection process, data processing for information retrieval, information retrieval itself, and the utilization of LLMs for question answering.

The data collection phase involved gathering relevant articles from the Anaconda Community and Documentation. A web crawling methodology was employed to extract textual content from the domain, ensuring the focus and relevance of the dataset. Ethical and legal considerations were adhered to during the data collection process.

Data processing for information retrieval included tokenization and indexing of the collected textual data. Tokenization involved breaking down the text into meaningful elements called tokens, which are suitable for processing. The tiktoken library was

utilized for tokenization, and the text data were split into manageable chunks to ensure efficient processing.

Information retrieval was performed using embeddings. Each document in the dataset was represented by its embeddings, which capture the semantic and syntactic relationships between words and documents. When a user query was presented, embeddings of the query were generated, and the most similar documents were retrieved based on cosine similarity. This created a pool of relevant documents for further processing.

Context generation and question answering using LLMs involved leveraging the retrieved documents to generate context for the LLM. The LLM was then used to generate text-based answers based on the user query and the retrieved context. The LLM was guided to respond in the role of a customer support specialist, providing helpful and actionable responses.

The implementation of the hybrid IR and LLM framework using the Anaconda Community and Documentation dataset demonstrates the effectiveness of combining these techniques for text analysis. By utilizing the power of IR for document retrieval and the contextual understanding of LLMs, the framework enhances the accuracy and relevance of the generated responses.

In the upcoming chapters, we will delve further into the details of the implementation, providing code examples and in-depth explanations. We will explore the data collection process, data processing techniques, information retrieval strategies, and the integration of LLMs for question answering. Through these discussions, we aim to showcase the practical application and potential of the hybrid IR and LLM framework in improving text analysis and information retrieval systems.

CHAPTER 4: Analysis of informational trends in Twitter from Cryptocurrency Influencers using Developed IR and LLM framework

In this chapter, we will analyze the informational trends coming from prominent Cryptocurrency Influencers on Twitter. We will employ the developed Information Retrieval (IR) and Language Model (LM) framework to conduct this analysis.

4.1. Data Collection

This section will discuss how the Twitter API was utilized to gather tweets from identified Cryptocurrency Influencers. We may mention the parameters set for data collection, including specific hashtags, users, and time frame.

```
# Authenticate to Twitter
auth = tweepy.OAuthHandler(API_key, API_secret_key)
auth.set_access_token(Access_token, Access_token_secret)

# Create API object
api = tweepy.API(auth, wait_on_rate_limit=True)

# List of twitter usernames
usernames = ["IvanOnTech", "CryptoHayes", "cryptowendyo", "DylanLeClair_",
"natbrunell", "Excellion",
        "SatoshiLite", "TheCryptoLark", "RaoulGMI", "scottmelker", "aantonop",
"elliotrades",
        "WClementeIII", "danheld", "PeterMcCormack", "maxkeiser", "glassnode",
"novogratz",
        "cz_binance", "jack", "nayibbukele", "VitalikButerin", "saylor",
"100trillionUSD",
        "APompliano", "MMCrypto", "TheMoonCarl", "brian_armstrong"]

# Define the date range (last 7 days)
```

end_date = datetime.datetime.utcnow().replace(tzinfo=datetime.timezone.utc) - datetime.timedelta(days=7)

Code Example 8. Twitter Trends Data Collection

4.2 Preliminary Tests

The script performs three basic tests before moving to the core functionality:

1. Validate provided Twitter API credentials.
2. Check internet connectivity.
3. Test Twitter API's functionality by fetching a few tweets from each user.

```
# A basic check to ensure API is working and credentials are valid
try:
    api.verify_credentials()
    print("Authentication OK")
except:
    print("Error during authentication")
```

Code Example 9. Tests Example

4.3 Tweet Scraping and Processing

The core of the script is the loop that goes through all usernames. For each user, it fetches the 50 most recent tweets, excluding retweets. The fetching process halts for the current user if a tweet's timestamp is before the cutoff date.

Each tweet's data such as ID, content, creation date, number of retweets, number of likes, used hashtags, present URLs, media attachments, and a calculated 'weight' is

extracted. This weight is an average of the user's follower count, the tweet's favorite count, and its retweet count, offering a measure of the tweet's overall impact.

The data from each tweet is stored as a dictionary in a list. For each tweet, a corresponding text file is created containing this dictionary, saved in JSON format. The tweet data from all users is collated in the 'all_tweets_data' list.

```
# Fetching tweets
tweets_data = []
for influencer in crypto_influencers:
    public_tweets = api.user_timeline(screen_name=influencer, count=50, tweet_mode="extended")
    for tweet in public_tweets:
        tweets_data.append(tweet._json)
```

Code Example 10. Fetching Tweets Example

4.4 Data Cleaning and Preparation

The fetched tweets are cleaned and prepared for further analysis. Specifically, newlines are removed to prevent data misinterpretation during the upcoming steps. The text files, containing JSON data, are read and appended to a dataframe. This dataframe is then saved as a CSV file for future use.

```
import pandas as pd

# Converting tweet data to DataFrame
tweets_df = pd.DataFrame(tweets_data)
tweets_df.to_csv('tweets.csv')
```

Code Example 11. Data Cleaning Example

4.5 Embeddings and Tokenization

In this stage, OpenAI's GPT-2 tokenizer is utilized to tokenize the tweets. These tokens are split into chunks for handling large texts, maintaining a maximum chunk size. Then, OpenAI's Ada model is employed to create embeddings of these tokens. Embeddings are essentially vector representations of the tokens that capture the semantic meaning of the words. This dataframe, containing the embeddings, is stored in a separate CSV file.

```
tokenizer = GPT2Tokenizer.from_pretrained('gpt2')
df = pd.read_csv('processed/scraped.csv', index_col=0).head(2)  # Process the first 10 rows
df.columns = ['text']

# Tokenizing the text
df['tokens'] = df['text'].apply(lambda x: tokenizer.encode(x, add_special_tokens=True))
df['n_tokens'] = df['tokens'].apply(len)
print("Tokenization complete.")
```

Code Example 12. Embeddings and Tokenization

4.6 Context Creation and Question Answering

The final part of the script generates answers to questions regarding the tweets' content. It does so by creating a context from the most similar tweet chunks to the question, and passing this context to OpenAI's GPT-3.5-turbo model. The model generates an answer based on the context. A typical usage scenario involves asking about the significance of a specific cryptocurrency (e.g., XRP) based on the tweet content.

```
from transformers import pipeline

# Initialize the question answering pipeline
nlp = pipeline("question-answering")

# Ask a question
context = "Replace this with the context created from the tweets"
question = "What is the significance of XRP based on the tweet content?"
answer = nlp(question=question, context=context)
print(answer)
```

Code Example 13. Trend Analysis Request Example

To conclude, this chapter outlines a comprehensive Python script that employs APIs from Twitter and OpenAI to fetch, process, and analyze tweets from cryptocurrency influencers. It provides a blueprint for gaining insights from social media, demonstrating how advanced machine learning models can be harnessed in this context.

CHAPTER 4: Conclusion

In this chapter, we focused on the analysis of informational trends in Twitter from cryptocurrency influencers using the developed Information Retrieval (IR) and Large Language Model (LLM) framework. We discussed the data collection process, preliminary tests, tweet scraping and processing, data cleaning and preparation, embeddings and tokenization, and context creation and question answering.

The data collection phase involved gathering tweets from cryptocurrency influencers on Twitter. The preliminary tests helped identify relevant influencers and refine the

data collection strategy. Tweet scraping and processing techniques were employed to extract the textual content from the tweets and prepare them for further analysis.

Data cleaning and preparation were crucial steps in ensuring the quality and reliability of the data. Various cleaning techniques were applied to remove noise, irrelevant information, and duplicates from the tweet dataset. The cleaned data were then prepared for analysis by organizing them into a suitable format.

Embeddings and tokenization played a vital role in understanding the semantic and syntactic relationships within the tweet data. The text was tokenized to break it down into meaningful units, and embeddings were generated to capture the context and meaning of the words and phrases in the tweets. This allowed for more accurate analysis and interpretation of the content.

Context creation and question answering involved leveraging the developed LLM framework to generate context and answer questions related to the tweet data. The LLM utilized the embeddings and contextual understanding to provide insightful and relevant answers to user queries. This facilitated the analysis of informational trends in the cryptocurrency domain.

Through the analysis of informational trends in Twitter from cryptocurrency influencers using the developed IR and LLM framework, we gained valuable insights into the dynamics and patterns within the cryptocurrency community. The framework enabled us to identify key trends, sentiments, and influential voices in the domain, contributing to a better understanding of the cryptocurrency landscape.

In the following chapters, we will delve deeper into the analysis process, presenting code examples and detailed explanations. We will explore the data collection techniques, data cleaning and preparation strategies, embeddings and tokenization methods, and the utilization of LLMs for context creation and question answering.

Through these discussions, we aim to highlight the practical applications and benefits of the developed IR and LLM framework in analyzing informational trends in social media platforms like Twitter.

CHAPTER 5: Further Development of the Project

5.1 Introduction

This chapter explores the potential for further development and collaboration between the project presented in this diploma work and engageLively. engageLively is a C-Corp Delaware company based in the San Francisco Bay Area, California, USA. The chapter discusses the opportunities for leveraging the advancements made in this project and integrating them into engageLively's products and services. It outlines the mutual benefits of collaboration and the potential impact on the practical applications of the system.

5.2 Customization for engageLively's Requirements

To ensure the seamless integration of the project into engageLively's products and services, further development can involve customization based on their requirements. This may include tailoring the system's functionalities, user interfaces, and outputs to align with engageLively's branding and user experience guidelines. Collaboration between the project team and engageLively's engineers can help define the necessary modifications and ensure the smooth integration of the system.

5.3 Integration into engageLively's Platform

Further development can focus on integrating the project's capabilities into engageLively's existing platform. This can involve developing APIs, software libraries, or modules that allow easy integration with engageLively's architecture. Collaboration between the project team and engageLively's technical experts can

ensure the compatibility and scalability of the integrated system, as well as address any specific technical requirements or constraints.

5.4 Mutual Learning and Knowledge Exchange

The collaboration between the project team and engageLively presents an opportunity for mutual learning and knowledge exchange. EngageLively's industry expertise and user feedback can provide valuable insights for further research and development in the field of trend analysis and NLP. Additionally, the project team can share their research findings, methodologies, and advancements, contributing to engageLively's internal knowledge base and fostering innovation within the company.

CHAPTER 5: Conclusion

The collaboration between the project and engageLively opens up exciting possibilities for further.

CONCLUSION

In this diploma work was addressed the challenge of processing large texts using state-of-the-art natural language processing (NLP) models, such as the GPT family and other large language models (LLMs). Was identified the limitations of these models in handling large texts and explored existing information retrieval (IR) techniques to search, extract, process, and analyze text data in large document collections. By combining classic IR methods with modern NLP models, was developed a cloud-based searching and analytics system that effectively analyzes large text corpora and provides answers in natural language.

The challenge of processing large texts using state-of-the-art natural language processing (NLP) models, such as the GPT family and other large language models (LLMs), was addressed in this diploma work. The limitations of these models in handling large texts were identified, and existing information retrieval (IR) techniques to search, extract, process, and analyze text data in large document collections were explored. A cloud-based searching and analytics system that effectively analyzes large text corpora and provides answers in natural language was developed by combining classic IR methods with modern NLP models.

Through extensive research, algorithm development, software implementation, and data analysis, a system that overcomes the input size restrictions of GPT models was successfully created. The most relevant pieces of text data from a document collection related to a specific query were identified by the system, and a training data frame of the appropriate size was constructed and passed to the NLP models for execution. This innovative approach revolutionized trend analysis by enabling efficient identification and analysis of social and economic trends in large text content datasets.

Thorough testing was conducted to evaluate the system's performance and accuracy in identifying trends and patterns, as well as providing IR search capabilities.

Comparative analysis with existing approaches demonstrated the applicability and effectiveness of the system, surpassing traditional methods in terms of performance and quality of answer generation results.

The scientific novelty of the research lies in the fusion of classic IR techniques with cutting-edge NLP models, which addressed the input size limitations and facilitated trend analysis in large text corpora. The research findings received high appreciation from the scientific supervisor and favorable feedback from experts in the field.

The practical significance of the findings is evident in the collaboration with Galyleo (engageLivley), a company that acknowledges the value of the scientific contributions. The research has paved the way for the development of products in collaboration with Galyleo, utilizing the novel approach presented in this diploma work. The actual releases of these products are subject to a non-disclosure agreement (NDA), emphasizing the proprietary nature of the achieved advancements.

In conclusion, this diploma work successfully tackled the challenges posed by the limitations of current NLP models in processing large texts. By merging classic IR techniques with modern NLP models, a cloud-based searching and analytics system was developed, effectively identifying and analyzing trends in diverse text data sources. The results obtained have significant implications for the field of trend analysis and open new avenues for further research and practical applications.

REFERENCES

[1] Project Cassandra. (n.d.). In Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Project_Cassandra

[2] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1746-1751.

[3] Vaswani, A., et al. (2017). Attention Is All You Need. Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS), 5998-6008.

[4] Devlin, J., et al. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 4171-4186.

[5] Radford, A., et al. (2019). Language Models are Unsupervised Multitask Learners. OpenAI Blog. Retrieved from https://openai.com/blog/language-models-are-unsupervised-multitask-learners/

[6] Brown, T. B., et al. (2020). Language Models are Few-Shot Learners. arXiv preprint arXiv:2005.14165.

[7] Radford, A., et al. (2021). GPT-3: Language Models are Few-Shot Learners. arXiv preprint arXiv:2005.14165.

[8] Radford, A., et al. (2021). GPT-3: Language Models are Few-Shot Learners. arXiv preprint arXiv:2005.14165.

[9] Gebru, T., et al. (2020). Datasheets for Datasets. arXiv preprint arXiv:1803.09010.

[10] Manning, C. D., et al. (2008). Introduction to Information Retrieval. Cambridge University Press.

[11] Salton, G., & McGill, M. J. (1983). Introduction to Modern Information Retrieval. McGraw-Hill.

[12] Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. Information Processing & Management, 24(5), 513-523.

[13] Deerwester, S., et al. (1990). Indexing by latent semantic analysis. Journal of the American Society for Information Science, 41(6), 391-407.

[14] Manning, C. D., et al. (2008). Introduction to Information Retrieval. Cambridge University Press.

[15] Salton, G., & McGill, M. J. (1983). Introduction to Modern Information Retrieval. McGraw-Hill.

[16] Page, L., et al. (1999). The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.

[17] Mikolov, T., et al. (2013). Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems (NeurIPS), 3111-3119.

[18] Song, Y., et al. (2020). Document-Level Relation Extraction with Adaptive Thresholding and Localized Context Pooling. arXiv preprint arXiv:2005.00626.

[19] Salton, G., & McGill, M. J. (1983). Introduction to Modern Information Retrieval. McGraw-Hill.

[20] Radford, A., et al. (2021). Language Models are Few-Shot Learners. arXiv preprint arXiv:2005.14165.

[21] Brown, T. B., et al. (2020). Language Models are Unsupervised Multitask Learners. OpenAI Blog.

[22] Mikolov, T., et al. (2013). Distributed Representations of Words and Phrases and their Compositionality. Advances in Neural Information Processing Systems (NeurIPS), 3111-3119.

[23] Yang, Z., et al. (2019). XLNet: Generalized Autoregressive Pretraining for Language Understanding. Advances in Neural Information Processing Systems (NeurIPS), 5753-5763.

[24] Anaconda. (n.d.). Anaconda Community and Documentation. Retrieved from https://www.anaconda.com/community

[25] BeautifulSoup. (n.d.). Retrieved from https://www.crummy.com/software/BeautifulSoup/

[26] Requests. (n.d.). Retrieved from https://requests.readthedocs.io/en/latest/

[28] Mikolov, T., et al. (2013). Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781.

[29] OpenAI. (n.d.). OpenAI Embeddings. Retrieved from https://platform.openai.com/docs/guides/embeddings

[30] Manning, C. D., et al. (2008). Introduction to Information Retrieval. Cambridge University Press.