T.R.
SAKARYA UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

FLEXIGPT: ENGAGING WITH DOCUMENTS

MSc THESIS

Abdalrhman ALQUAARY

Information Systems Engineering Department

JANUARY 2024

T.R.
SAKARYA UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

FLEXIGPT: ENGAGING WITH DOCUMENTS

MSc THESIS

Abdalrhman ALQUAARY

Information Systems Engineering Department

Thesis Advisor: Prof. Dr. Numan Çelebi

JANUARY 2024

The thesis work titled “FLEXIGPT: ENGAGING WITH DOCUMENTS” prepared by Abdalrhman Alquaary was accepted by the following jury on 17/01/2024 by unanimously/majority of votes as a MSc THESIS in Sakarya University Graduate School of Natural and Applied Sciences, Information Systems Engineering department.

Head of Jury :

Jury Member :

Jury Member :

Thesis Jury

STATEMENT OF COMPLIANCE WITH THE ETHICAL PRINCIPLES AND RULES

I declare that the thesis work titled "FLEXIGPT: ENGAGING WITH DOCUMENTS", which I have prepared in accordance with Sakarya University Graduate School of Natural and Applied Sciences regulations and Higher Education Institutions Scientific Research and Publication Ethics Directive, belongs to me, is an original work, I have acted in accordance with the regulations and directives mentioned above at all stages of my study, I did not get the innovations and results contained in the thesis from anywhere else, I duly cited the references for the works I used in my thesis, I did not submit this thesis to another scientific committee for academic purposes and to obtain a title, in accordance with the articles 9/2 and 22/2 of the Sakarya University Graduate Education and Training Regulation published in the Official Gazette dated 20.04.2016, I accept all kinds of legal responsibility that may arise in case of a situation contrary to this statement.

(04/12/2023)

(signature)

Abdalrhman ALQUAARY

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my family, especially my parents, for their unwavering support and encouragement throughout my academic journey. Their love, understanding, and sacrifices have been the cornerstone of my achievements. I am particularly indebted to my brother Alaa Aldean for his help and to my other brothers, whose support and belief in me have been endless. I owe them a debt of gratitude that words can scarcely express.

I am immensely grateful to my Thesis Advisor, Prof. Dr. Numan Çelebi, for his invaluable guidance, patience, and expertise. His mentorship has been instrumental in shaping not only this thesis but also my approach to scholarly inquiry and research. His dedication to my academic growth has been a source of inspiration and motivation.

I would also like to extend my heartfelt thanks to the staff of the Information Systems Engineering Department At Sakarya University. Their collective knowledge, support, and encouragement have created an environment that is both challenging and nurturing, fostering my intellectual growth and professional development.

This journey would not have been possible without the collective support of these individuals and many others. I am deeply thankful to each one of them for their contribution to my academic endeavor.

Abdalrhman ALQUAARY

CONTENTS

Page

ABBREVIATIONS

AI	**:** Artificial Intelligence
BERT	: Bidirectional Encoder Representations from Transformers
CPU	: Central Processing Unit
CLI	: Command Line Interface
GGML	: GPT-Generated Model Language
GPT	: Generative Pre-trained Transformer
GPTQ	: Generative Pre-trained Transformers Quantization
GPU	: Graphics Processing Unit
GUI	: Graphical User Interface
LLM	: Large Language Model
LoRA	: Low-Rank Adaptation
NLP	: Natural Language Processing
QLoRA	: Quantized Low-Rank Adaptation
RAG	: Retrieval-Augmented Generation

LIST OF FIGURES

Page

LIST OF TABLES

Page

FLEXIGPT: ENGAGING WITH DOCUMENTS

SUMMARY

In the rapidly evolving domain of AI, the emergence of LLMs has catalyzed a shift across various sectors, fundamentally reshaping the dynamics of digital interaction. Amidst this technological renaissance, our FlexiGPT emerges as a groundbreaking application, leveraging the full potential of these advancements. Designed as a CLI program, FlexiGPT serves as a bridge between users and their digital files, enabling a high level of interactive engagement.

At the heart of FlexiGPT's innovation is its ability to download and integrate any LLM or embedding model from the Hugging Face platform. This feature empowers users to tap into a diverse array of the latest language models, ensuring they remain at the forefront of linguistic technology. The integration process is streamlined and user-friendly, making advanced AI tools accessible to a broader audience.

FlexiGPT employs the RAG technique, a sophisticated approach that combines LLMs with advanced retrieval methods. This amalgamation allows the program to not only generate relevant responses but also to show the data sources that it used to answer the queries, enhancing the depth and accuracy of interactions.Users can query their digital files, receive insightful responses, and engage in a dialogue that was once the domain of science fiction.

The flexibility of FlexiGPT extends beyond its model integration capabilities. Users can choose to load models in their full version for maximum performance or quantized forms and specialized formats like GPTQ or GGML, balancing performance with resource requirements. This adaptability makes FlexiGPT a versatile tool, suitable for various applications, from academic research to business analytics

Another feature of FlexiGPT is its local execution capability. By running the program on local machines, users benefit from enhanced data privacy and security. This local approach also allows for faster response times and reduced reliance on cloud services, which is particularly advantageous in scenarios where internet connectivity is limited or unreliable.

FlexiGPT's impact extends beyond its technical prowess. The application opens new avenues for how individuals and organizations interact with their digital data. From simplifying complex data analysis to enabling more natural and efficient file management, FlexiGPT stands as a testament to the practical applications of AI in everyday life.

FlexiGPT represents a significant leap forward in the field of AI and digital interaction. By harnessing the power of the latest language models and embedding them within an accessible and versatile interface, FlexiGPT is not just a tool; it is a harbinger of a new era in digital communication and interaction. As technology continues to evolve, FlexiGPT is poised to adapt and grow, offering users an ever-expanding suite of capabilities to explore the full potential of AI-driven digital engagement.

FLEXIGPT: BELGELERLE ETKİLEŞİM

ÖZET

Yapay zekanın hızla gelişimi ile birlikte Büyük Dil Modelleri (LLM'ler) olarak adlandırılan teknolojiler ortaya çıkışmıştır. Bu teknolojiler bir çok sektörlerde değişimlere yol açmıştır. Ayrıca mevcut dijital etkileşimin temel dinamiklerini önemli ölçüde değiştirmiştir. Bu teknolojik gelişmelere dayanarak bu çalışmada FlexiGPT adlı bir uygulama geliştirilmiştir. Bu uygulama, bu teknolojilerin potansiyelini koruyarak, komut satırı arayüzü (CLI) programı olarak tasarlanmıştır. Geliştirilen program kullanıcılar ile dijital dosyaları arasında esnek bir etkileşim düzeyi sağlamaktadır. FlexiGPT, kullanıcıların dijital içeriklerle olan etkileşim kurma biçimine yeni bir yaklaşım getirirken aynı zamanda yapay zeka teknolojilerinin sahip olduğu geniş imkanları da kullanıcılara sunmaktadır. FlexiGPT, kullanıcılar ve verileri arasında güçlü bağlantı kurma yeteneği sayesinde büyük dil modellerinin kullanımı için yeni bir bakış açısı oluşturmaktadır. Bu program, kullanıcı deneyimini zenginleştirmenin yanı sıra, dijital dosyaların kullanılabilirliğini de maksimize etmeyi amaçlamaktadır.

FlexiGPT'nin yenilikçilik merkezinde, Hugging Face platformundan herhangi bir LLM veya gömülü modeli indirme ve entegre etme kapasitesi yer almaktadır. Bu özellik, kullanıcılara, dil teknolojisi alanındaki en son gelişmelere erişim imkanı sunarak, onları bu alandaki yeniliklerin ön saflarında tutmaktadır. Entegrasyon süreci, son derece akıcı ve kullanıcı dostu olarak tasarlanmıştır. FlexiGPT bu özelliği ile, gelişmiş yapay zeka araçlarını daha geniş bir kitleye eriştirme imkanı vermektedir. Dil modellemesinin son yeniliklerini kullanarak, FlexiGPT, metin işleme, anlama ve etkileşimde yeni bir yaklaşım getirmektedir. Bu, çok yönlü ve gelişmiş dil modelleri arasından seçim yapabilme özgürlüğü ile birleştiğinde, kullanıcıların bireysel ihtiyaçlarına ve özel senaryolarına göre özelleştirilebilir bir ortam sunar. FlexiGPT, bu yenilikçi yaklaşımıyla, dil işleme teknolojilerinin kapsamını genişleterek, kullanıcıların dijital içerikle etkileşim şeklini yeniden tanımlamakta ve böylece yapay zekanın dil anlayışını ve üretimini, gerçek dünya uygulamalarıyla bütünleştirmektedir.

FlexiGPT, geleneksel dil modellerini ileri düzey bilgi alma yöntemleriyle birleştiren sofistike bir yaklaşım olan RAG tekniğini kullanmaktadır. Bu entegrasyon, programın sadece ilgili yanıtlar üretmesini sağlamakla kalmaz, aynı zamanda sorgulara yanıt verirken kullandığı veri kaynaklarını göstererek etkileşimlerin derinliğini ve doğruluğunu artırır. Kullanıcılar, dijital dosyalarını sorgulayabilir, içgörülü yanıtlar alabilir ve bir zamanlar bilim kurgunun alanı olarak kabul edilen bir diyalog başlatabilirler. FlexiGPT'nin bu yeteneği, yapay zekanın sadece veri işleme ve yanıt üretme kabiliyetini değil, aynı zamanda anlamayı ve öğrenmeyi de içeren çok daha geniş bir potansiyeli ortaya koymaktadır. Bu teknik sayesinde, kullanıcılar sadece zengin ve katmanlı bilgiler elde etmekle kalmaz, aynı zamanda programın cevaplarını oluştururken hangi bilgi kaynaklarından yararlandığını anlama fırsatı bulurlar. Buda verilen yanıtların şeffaflığını ve güvenilirliğini artırmaktadır.

FlexiGPT'nin esnekliği, sadece model entegrasyon yetenekleri ile sınırlı değildir; aynı zamanda kullanıcıların özelleştirilmiş ihtiyaçlarına uygun çözümler sunma yeteneği ile de dikkat çeker. Kullanıcılar, FlexiGPT içinde sunulan geniş model yelpazesi arasından seçim yaparak, projelerinin gereksinimlerine en iyi şekilde uygun olan modeli seçme özgürlüğüne sahiptirler. Tam sürüm modeller, en yüksek performansı elde etmek isteyenler için idealdir, bu sayede karmaşık veri analizlerini veya büyük ölçekli iş analitiklerini sorunsuz bir şekilde gerçekleştirebilirler. Ancak, kaynak gereksinimlerini optimize etmek veya belirli senaryolara uyum sağlamak isteyenler için, kuantize edilmiş formlar veya özel GPTQ veya GGML formatları gibi alternatif seçenekler de mevcuttur. Bu uyumluluk, FlexiGPT'yi akademik araştırmalardan iş analitiklerine kadar geniş bir yelpazede uygulamalara uygun bir araç haline getirir. Kullanıcıların projelerine en iyi şekilde hizmet edebilmesi için özelleştirilebilir bir yapı sunarak, çeşitli performans ve kaynak gereksinimlerine mükemmel bir şekilde uyan çözümler sunar. Bu sayede, her türlü projenin gereksinimlerine uygun bir FlexiGPT konfigürasyonu oluşturmak, kullanıcıların elindedir.

FlexiGPT'nin bir diğer önemli özelliği, yerel çalıştırma kapasitesidir. Programı yerel makinelerde çalıştırarak, kullanıcılar gelişmiş veri gizliliği ve güvenliğinden yararlanır. Bu yerel yaklaşım, aynı zamanda daha hızlı yanıt süreleri sağlar ve bulut hizmetlerine olan bağımlılığı azaltır. Bu yapı, özellikle internet bağlantısının sınırlı veya güvenilmez olduğu durumlarda büyük avantaj sağlar. Yerel çalıştırma yeteneği, kullanıcıların kendi donanımlarını kullanarak FlexiGPT'nin tüm özelliklerinden faydalanmalarını sağlar. Bu yetenek, özellikle veri hassasiyeti yüksek olan projelerde büyük önem taşır. Ayrıca, yerel çalıştırma, veri işleme ve analiz süreçlerinde daha fazla kontrol ve esneklik sunar, buda kullanıcıların özel gereksinimlerine ve çalışma koşullarına daha iyi uyum sağlamalarına olanak tanır.

FlexiGPT, bireylerin ve kuruluşların dijital verilerle etkileşim kurma yöntemlerinde yeni kapılar aralamaktadır. Sadece karmaşık veri analizlerini basitleştirmekle kalmaz, aynı zamanda daha doğal ve verimli dosya yönetimine olanak tanır. Bu sayede yapay zekanın günlük yaşamdaki somut uygulamalarına dair çarpıcı bir örnek sunar. FlexiGPT'nin bu geniş kapsamlı etkisi, yapay zekanın yalnızca akademik ve profesyonel alanlarla sınırlı olmadığını, aynı zamanda günlük yaşamın her alanında değerli bir rol oynayabileceğini göstermektedir. Ayrıca, toplumda yapay zekanın yaygınlaşmasına büyük katkı sağlayan bir uygulama özelliği taşımaktadır. Yapay zeka teknolojilerinin yaygınlaşması, günlük yaşamımızı dönüştürerek, her seviyede kullanıcı için daha erişilebilir ve kullanıcı dostu hale getirir. FlexiGPT, bu değişimdeki öncü rolü ile, dijital etkileşim ve yapay zeka kullanımının geleceğini şekillendirmeye yardımcı oluyor. Bu sayede, iş dünyasından eğitime, sanat ve eğlenceden sağlık hizmetlerine kadar birçok farklı alanda çeşitli uygulamaların önünü açıyor ve kullanıcıların dijital dünyayı daha etkili bir şekilde deneyimlemesine olanak tanıyor.

FlexiGPT, yapay zeka ve dijital etkileşim alanlarında kaydedilen ilerlemeler arasında önemli bir adımı temsil etmektedir. En son dil modellerinin gücünü kullanarak ve bu modelleri erişilebilir, çok yönlü bir arayüz içinde entegre ederek, FlexiGPT sadece bir araç olmanın dışında, dijital iletişim ve etkileşimde yeni bir bakış açısı getirmiştir. Teknoloji alanındaki sürekli gelişmelerle birlikte, FlexiGPT'nin de sürekli olarak adapte olması ve büyümesine ihtiyaç vardır. FlexiGPT, yapay zekanın sadece veri işleme ve dil anlama yeteneklerine dayanmayıp, bu teknolojileri kullanıcıların günlük etkileşimlerine entegre ederek, dijital dünyada daha zengin ve anlamlı deneyimler oluşturmak için yeni yollar açmaktadır. Bu gelişme, yapay zeka ve dil modellerinin, geniş bir kullanım alanına sahip pratik ve etkili araçlar olarak kullanımını yaygınlaştırırken, dijital etkileşim alanında yeni standartlar belirlemekte ve geleceğin teknolojik manzarasını şekillendirmektedir. FlexiGPT'nin bu yenilikçi ve esnek yapısı, yapay zeka tabanlı dijital etkileşimin, hem iş dünyasında hem de günlük yaşamda nasıl bir dönüşüm yaratabileceğine dair önemli bir örnek teşkil etmektedir.

1. INTRODUCTION

In this section, we lay the groundwork for our work by delving into the general concepts fundamental to understanding our study. This segment is designed to provide a comprehensive overview of the key principles, theories, and frameworks that form the basis of our research. It serves as a primer, equipping the reader with the necessary background and context to fully appreciate the nuances and intricacies of the topics discussed in subsequent chapters. This foundational knowledge is crucial for a holistic understanding of the advanced concepts and methodologies that will be examined in detail as we progress through the thesis.

1.1. Natural Language Processing

In the ever-evolving landscape of Artificial Intelligence (AI), one area has emerged as a powerful catalyst for change: Natrual Language Processing (NLP). NLP is not merely a technological endeavor; it represents a profound shift in how machines understand and interact with human language, opening doors to unprecedented possibilities. At its core, NLP is the interdisciplinary field that bridges the intricacies of human language with the computational power of machines. It seeks to equip computers with the ability to comprehend, interpret, and generate human-like text, thereby facilitating meaningful communication between humans and machines.

The essence of NLP lies in its capacity to unravel the complexity of language, encompassing not just the syntactic structure but also the semantics and pragmatics that give words and sentences their nuanced meaning. By delving into the subtle nuances of language, NLP enables machines to go beyond mere word recognition and engage in a more profound understanding of context, intent, and sentiment. This capability has profound implications across various domains, ranging from customer service interactions and virtual assistants to sentiment analysis in social media, advanced language translation and text generation.

In its quest to bridge the gap between human language and machines, NLP draws upon a diverse array of techniques and methodologies, including machine learning, deep learning, and linguistic theories. These tools empower algorithms to discern patterns, learn from vast datasets, and continuously refine their language-processing capabilities. Sentiment analysis, named entity recognition, and machine translation are just a few examples of the practical applications of NLP that have become integral parts of our daily lives.

Furthermore, the advent of pre-trained language models, such as Generative Pre-trained Transformer (GPT-3), has ushered in a new era in NLP, where models can understand and generate human-like text with remarkable fluency and coherence. These models, trained on massive datasets, have the ability to generalize their understanding across a wide range of topics and contexts, enabling them to adapt to diverse language tasks with minimal fine-tuning.

As NLP continues to advance, its impact on society is profound. Beyond enhancing the efficiency of human-computer interactions, NLP contributes to breaking down language barriers, fostering cross-cultural communication, and promoting inclusivity. It empowers applications to be more user-friendly, facilitating accessibility for individuals with diverse linguistic backgrounds. Moreover, the ethical dimensions of NLP, including concerns related to bias in language models and the responsible use of AI in decision-making processes, have sparked critical discussions within the scientific and societal realms.

1.2. Large Language Models

Within the dynamic and swiftly evolving realm of NLP, the emergence and proliferation of Large Language Models (LLMs) stands as a turning point. These models, characterized by expansive neural architectures and formidable parameter counts, have become a focal point of attention, driving significant advancements in the capabilities of AI systems to understand and generate human-like text.

The diffusion of LLMs across various domains is noteworthy, as their influence extends beyond the confines of traditional NLP research. These models, exemplified prominently by GPT-3, have found applications ranging from content creation and language translation to code generation and creative writing. Their widespread adoption underscores a paradigm shift in how machines interact with and process natural language, permeating diverse sectors and reshaping the landscape of AI applications.

1.3. Quantization

In the expansive domain of LLMs, where the execution of these models poses formidable computational challenges, the pursuit of computational efficiency and model deployment has given rise to a significant area of inquiry known as quantization. This technique, characterized by the process of reducing the precision of numerical representations within neural network parameters, holds paramount importance in the context of LLMs, particularly given the computational challenges inherent in their execution. Quantization is instrumental in addressing the formidable computational demands associated with the deployment of LLMs in real-world applications, offering a pathway towards enhanced efficiency without compromising the inherent linguistic capabilities of these models.

The exploration of quantization within the realm of LLMs signifies a nuanced convergence of theoretical considerations and practical implications. As LLMs continue to gain prominence in various applications, from natural language understanding to generation tasks, the imperative to optimize their computational footprint becomes increasingly evident. Quantization, as a methodological approach, presents a compelling avenue to strike a delicate balance between model efficiency and linguistic prowess, crucial in mitigating the challenges posed by the computational intensity inherent in running LLMs.

1.4. Prompt Engineering

Within the intricate domain of LLMs, the concept of prompt engineering emerges as a focal point of scholarly investigation. Prompt engineering, a nuanced and deliberate formulation of input queries or instructions, plays a crucial role in influencing the output and behavior of LLMs. This strategic approach stands as a response to the inherent challenges associated with effectively harnessing the vast linguistic capabilities of these models, thereby contributing to the enhacement of LLMs.

The impetus behind the exploration of prompt engineering lies in the acknowledgment of the intricacies and potential biases embedded in LLMs. As these models exhibit immense linguistic proficiency, they also possess the capacity to generate outputs that may inadvertently reflect or perpetuate existing societal biases. Prompt engineering, as a methodological intervention, seeks to refine and guide the language generation process, offering researchers and practitioners a means to shape the output of LLMs in a more controlled and intentional manner.

1.5. Embedding Models

Embedding models in NLP transform textual data into embedding vectors, numerical representations that capture semantic relationships and syntactic structures within the language. These vectors represent words or phrases as points in a multidimensional space, allowing for the quantification of linguistic similarities and differences. The conversion to embedding vectors is crucial for a variety of NLP tasks, enabling machines to process and understand human language in a meaningful and computationally efficient manner.

1.6. Retrieval Augmented Generation

In the expansive landscape of LLMs, the emergence of the Retrieval-Augmented Generation (RAG) paradigm marks a significant innovation, introducing a nuanced and multifaceted framework to the field. RAG marries retrieval-based mechanisms with the sophisticated generative capacities inherent in modern language models. This approach not only enhances the capabilities of LLMs but also enriches the quality of generated content by integrating contextual data from a myriad of external sources.

In the realms of AI and NLP, RAG signifies a cutting-edge frontier. It demonstrates an advanced amalgamation of computational techniques, which extends beyond basic language understanding and generation. By effectively sourcing and incorporating relevant external information, RAG-equipped LLMs achieve a higher level of accuracy and context-awareness in their responses. This is especially pivotal in tasks that demand nuanced understanding and up-to-date knowledge, such as answering complex queries, content creation, and dynamic text generation.

RAG's potential in enhancing the interactivity and adaptability of LLMs is profound. It allows these models to transcend traditional boundaries, moving from static text generation to creating outputs that reflect current, real-world information and nuanced understanding. This makes RAG an invaluable asset in applications where the synthesis of vast amounts of data is crucial, including in sectors like healthcare, finance, legal analysis, and customer service, where accurate and contextually rich responses are essential. The RAG framework, by bridging the gap between generative prowess and contextual relevance, is poised to redefine the standards of language models, pushing the envelope of what is achievable in the field of AI-driven language processing and interaction.

1.7. Semantic Search

Leveraging advanced language models and text embeddings, has significantly transformed the landscape of information retrieval, particularly in the context of RAG techniques. This approach enables a markedly more precise and efficient method for document retrieval. By understanding the deeper meaning and context embedded within queries, semantic search transcends traditional keyword-based methods, offering results that are not just relevant, but contextually aligned with the user's intent. This evolution in search technology marks a substantial leap forward in our ability to access and utilize information effectively.

The incorporation of Semantic Search within the purview of LLMs signifies a notable convergence of advanced computational techniques, reflecting a purposeful effort to transcend traditional keyword-based search methodologies. In the expansive landscape of natural language processing and artificial intelligence, LLMs have demonstrated unparalleled proficiency in understanding and generating coherent textual content. However, the integration of Semantic Search introduces an additional layer of sophistication, aiming to refine the search process by incorporating a deeper understanding of the meaning, context, and relationships embedded within textual data.

At its core, Semantic Search epitomizes a distinctive paradigm in information retrieval. It departs from conventional keyword matching approaches and seeks to comprehend the underlying semantics of user queries, enabling a more contextually relevant exploration of large datasets. By harnessing the semantic understanding encapsulated within LLMs, Semantic Search aspires to bridge the gap between user intent and search results, offering a more nuanced and contextually rich retrieval experience.

1.8. Hugging Face

Hugging Face has emerged as a leading figure in the realm of AI, particularly in the field of NLP. Renowned for its open-source approach, Hugging Face has democratized access to state-of-the-art language models and AI tools, fostering a collaborative environment for researchers, developers, and organizations worldwide (Hugging Face, 2023). Beyond providing access to these models, Hugging Face has developed an ecosystem that facilitates model training, sharing, and deployment, thus accelerating innovation and application in AI.

1.9. LangChain

LangChain, a novel framework in the realm of computational linguistics and artificial intelligence, marks a significant advancement in the application and utilization of LLMs. Developed to streamline the integration and interaction of LLMs in complex applications, LangChain addresses the growing need for modular and flexible systems in the rapidly evolving field of NLP (LangChain, 2023).

Fundamentally, LangChain is engineered to enable the construction of sophisticated language-based applications. It achieves this by providing a versatile framework that facilitates the orchestration of various components essential for advanced language processing tasks. These components include, but are not limited to, the LLMs themselves, retrieval mechanisms for sourcing information, reasoning modules, and intricate dialogue management systems. The framework is distinctively characterized by its modularity, allowing for seamless integration and interchange of different models and tools tailored to specific application requirements.

1.10. FlexiGPT Purpose and Importance

The purpose and importance of FlexiGPT program center around its role in facilitating meaningful and interactive communication with documents, leveraging the Retrieval-Augmented Generation (RAG) framework. This program is for enhancing the quality and relevance of generated answers while seamlessly integrating retrieval-based mechanisms and generative capabilities. Its significance lies in its ability to empower users to engage with textual content in a dynamic and contextually rich manner by utilizing AI-generated interactions, effectively bridging the gap between human intent and information retrieval. FlexiGPT promotes effective interactions with documents, ultimately contributing to more efficient and impactful utilization of textual data in various senarios, thereby allowing users to have a better experience in dealing with their textual files.

2. LITERATURE OVERVIEW

In this focused exploration of NLP, we aim to provide a succinct yet comprehensive view of its progression and breakthroughs, particularly through the lens of four key developments: the evolution of LLMs, the impact of quantization techniques, the advancements in embedding models, and the diverse applications of RAG, coupled with our research contributions.

The evolution of NLP, particularly through the development of transformer-based models, is well-documented in a series of groundbreaking papers. Following the seminal "Attention Is All You Need" by Vaswani et al. (2017), which introduced the transformer model, there has been a surge in significant research contributing to this domain. The introduction of GPT by Radford et al. (2018), marked a pivotal moment in NLP, showcasing the model's ability to generate coherent and contextually relevant text. The subsequent iterations, notably GPT-2 and GPT-3 by (Radford et al. 2019; Brown et al. 2020) achieved remarkable feats in advanced text generation. Bidirectional Encoder Representations from Transformers (BERT), introduced by Devlin et al. Devlin et al. (2018) represents another major advancement. BERT's bidirectional training revolutionized the way models understand contextual relationships in text, setting new standards in a variety of language tasks. Following this line of thought, Raffel et al. (2020) introduction of the T5 (Text-To-Text Transfer Transformer) model represents a significant advancement in NLP. It reimagines various NLP tasks as uniform text-to-text conversions, offering a versatile and unified framework for addressing diverse NLP challenges. Large Language Models (LLMs) have shown remarkable proficiency in arithmetic and symbolic reasoning tasks using few-shot prompting methods, including the "chain-of-thought" technique, which leverages LLMs for problem decomposition and solution. In their study, Penedo et al. (2023) introduced 'Falcon', a Large Language Model, which challenges conventional training approaches. This model illustrates that models trained solely on carefully filtered and deduplicated web data can surpass the performance of those trained on established high-quality datasets like The Pile (Gao et al, 2020).

Running LLMs presents significant challenges due to their high hardware costs and computational demands; the quantization technique in LLMs emerges as a strategy aimed at mitigating these issues, significantly enhancing model efficiency and scalability. Marking a breakthrough in the domain of LLMs, the Low-Rank Adaptation (LoRA) technique enables efficient fine-tuning of these models by introducing low-rank matrices, significantly reducing the computational burden and resource requirements (Hu et al, 2021). Building on the foundation set by the LoRA technique, the Quantized Low-Rank Adaptation (QLORA) approach, introduced by Dettmers et al. (2023), takes this concept a step further. It allows for the fine-tuning of large models, such as those with 65B parameters, on a single 48GB Graphics Processing Unit (GPU), effectively balancing memory usage and performance. By combining 4-bit quantized pretrained language models with LoRA, QLORA demonstrates its efficacy, outperforming existing models and offering significant innovations in memory management and model training efficiency. Addressing the substantial computational and storage demands of large-scale GPT models, Frantar (2023) developed Generative Pre-trained Transformers Quantization (GPTQ), a groundbreaking quantization approach. This method notably reduces model size by compressing to 3 or 4 bits per weight, effectively balancing the need for efficiency with minimal accuracy loss. In another study, Naveed et al. (2023) have meticulously compiled a comprehensive overview of recent advancements in LLMs, addressing a broad spectrum of topics including architectural innovations, efficiency improvements, and more. Their work not only provides detailed summaries and analyses of current models and datasets but also serves as an essential reference for the latest trends and insights in LLM research.

In the sphere of NLP, embedding models have revolutionized how machines understand and interpret human language, serving as a cornerstone for various advanced applications and research. Efficiently creating vector representations of words that capture their contextual meanings within large text corpora was significantly advanced by the introduction of Word2Vec (Mikolov et al, 2013). This technique marked a substantial progression in the field by its ability to generate meaningful word embeddings. Building on the advancements made by Word2Vec, Global Vectors for Word Representation (GloVe) introduced by Pennington et al. (2014), further enriched the field of word embeddings. GloVe employed a unique method that blends matrix factorization techniques with local context window approaches, capturing both global and local linguistic features in its embeddings. This development represented another significant step in the evolution of embedding models. The introduction of transformer models marked a significant advancement in the field of embedding models, leading to substantial improvements in their performance and capabilities. Addressing the limitations of both general-purpose and task-specific retrievers, the LLM-Embedder developed by Zhang et al. (2023) represents a significant advancement in retrieval augmentation for LLMs. Their innovative training strategies, such as LLM feedback-based optimization and multi-task fine-tuning, have shown marked improvements in various retrieval scenarios. Asudandi et al. (2023) provide a thorough review of word embedding and deep learning models, focusing on their application in natural language processing tasks. Their study contrasts and compares various techniques, offering insights for selecting the most effective models for specific text analytics tasks.

The RAG framework, introduced in the landmark paper "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks" by Lewis et al. (2021), represents a development in NLP. This innovative framework integrates pre-trained parametric memory, such as seq2seq models, with non-parametric memory, notably a dense vector index of Wikipedia, thereby revolutionizing language generation tasks. RAG models excel in a variety of knowledge-intensive NLP tasks, especially in setting new benchmarks in open domain questioning-answering tasks, through their ability to generate specific, diverse, and factually accurate language. Building on this foundation, Melz (2023) further advanced the capabilities of RAG with the introduction of ARM-RAG, which enhances problem-solving in complex areas like grade-school math by efficiently using retrieval-augmented generation to store and access reasoning chains.

Amidst these advancements, this thesis introcues FlexiGPT. FlexiGPT emerges as a program that adds flexibility to the RAG framework. It enables users to choose from a wide array of models on platforms like Hugging Face, thereby customizing the NLP experience to meet diverse needs and objectives (Alquaary et al, 2023; FlexiGPT, 2023). FlexiGPT is a testament to the evolving synergy between user-centric design and cutting-edge NLP technologies, representing a significant leap forward in making

NLP frameworks more adaptable and aligned with the dynamic requirements of the field.

3. METHODOLOGY

In the methodology section of our study, we introduce FlexiGPT, an innovative Command Line Interface (CLI) program that incorporates the RAG framework to address the challenges of query-based information retrieval. A key feature of FlexiGPT is its user-driven design, which allows users to select both the embedding model and the LLM from the comprehensive range available on the Hugging Face platform. This unique capability offers unparalleled flexibility, enabling users to tailor the information retrieval process to their specific needs. FlexiGPT harnesses these chosen models to process and interpret complex queries across various file formats, thereby enhancing the precision and contextual relevance of the retrieved information. This approach represents a significant methodological shift in the application of NLP technologies, providing a more adaptable and efficient solution for extracting relevant information in response to diverse user queries. The integration of user-selected models in FlexiGPT not only demonstrates an advancement in NLP capabilities but also contributes a novel perspective to the field of intelligent information retrieval.

Figure 3.1 presents a simplified overview of the RAG process. It begins with a repository of documents, which serves as a knowledge base. When a query is made, the RAG system first engages in the retrieval phase, where relevant information is sourced from the documents. This information then flows into the generation phase, where the retrieved data is synthesized into a coherent answer. The final output is delivered to the user, who interacts with the system, possibly through a computer interface. The flow from documents to a user's answer encapsulates the essence of RAG, blending retrieval of information with generative AI to provide informed and accurate responses.

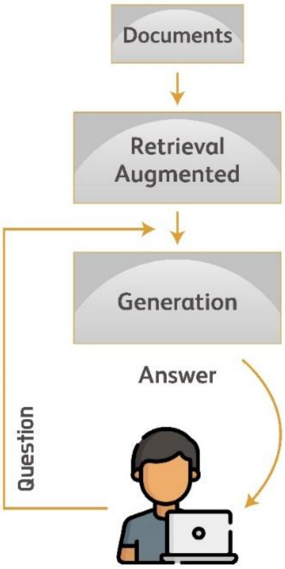

Figure 3.1. RAG Framework

3.1. FlexiGPT Process

The FlexiGPT program leverages the RAG technique, augmenting it with enhanced customization features that empower users to tailor the information retrieval and generation process. As depicted in the Figure 3.2, the FlexiGPT architecture begins with the extraction of data from a specified source, which is then segmented into discrete chunks. These chunks are processed through a user-selected embedding model, resulting in a series of embeddings that populate a vector database. Through semantic search, the most relevant embeddings are identified in response to a user-generated query, which are then utilized by a chosen LLM to generate an accurate and contextually relevant answer. This figure encapsulates the FlexiGPT's workflow, illustrating its modular design that enables users to select different embedding models and LLMs for various purposes.

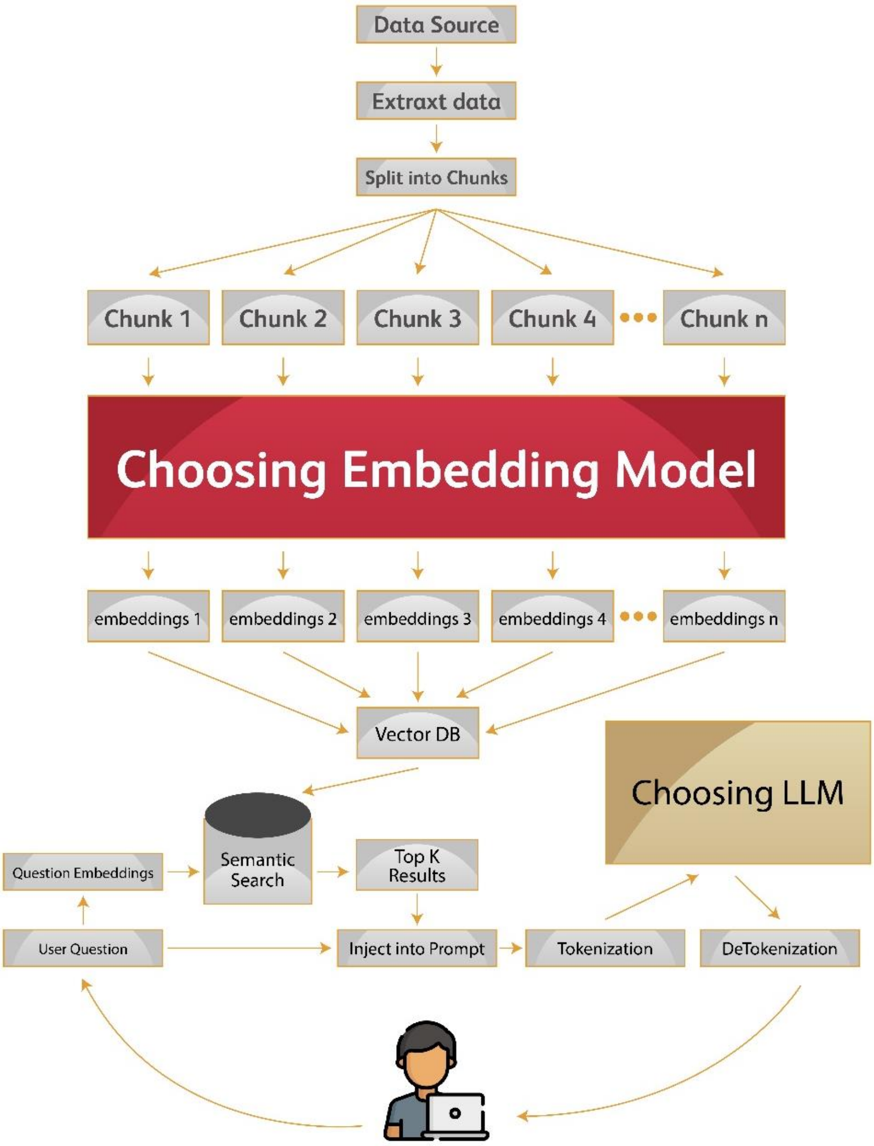

Figure 3.2. FlexiGPT Process

3.1.1. Data source and extraction

The FlexiGPT program initiates its process by sourcing documents, which may come in various formats such as TXT, DOCX, or PDF. Users can provide multiple files as input, facilitating a comprehensive knowledge base for the system to draw from. By default, FlexiGPT searches for these files within a 'docs' folder located in the program's directory. However, this pathway is not fixed; users have the flexibility to customize the file path, allowing them to direct FlexiGPT to alternative locations where their documents are stored. This adaptability ensures that the system can access the necessary information to address user queries from a diverse range of document repositories.

3.1.2. Chunking data

Once the FlexiGPT program has gathered the data from the provided documents, it divides the data into smaller sections, or 'chunks', to accommodate the input size limitations of LLMs. Each chunk contains 1000 characters, with a 200-character overlap between consecutive chunks to ensure no contextual information is lost. This method maintains the integrity and continuity of the data. Furthermore, each chunk is associated with metadata identifying its source file, adding a layer of traceability. The open-source nature of FlexiGPT allows users to customize the chunk size by modifying the code, providing the flexibility to tailor the data segmentation to the specific requirements of their LLMs and the tasks at hand. This chunking technique is a strategic approach to efficiently process and utilize large volumes of text within the constraints of LLM input capacities.

3.1.3. Embedding model selection

A key feature of the FlexiGPT program is the capacity it offers users to actively choose an embedding model that aligns with the nuanced requirements of their tasks (Embedding Models, 2023). This selection process, which taps into the extensive repository on the Hugging Face platform, is not just an additional feature but a cornerstone of the FlexiGPT's design philosophy, emphasizing user control and customization. When a user decides on a particular model, the FlexiGPT system efficiently checks its local availability. If the model is already installed on the user's system, it is immediately utilized; if not, the program conveniently handles the download, ensuring the model is promptly ready for operation.

This process is particularly user-friendly because it removes barriers to entry for those not familiar with manual model management. It also ensures that FlexiGPT remains accessible to a broader audience, ranging from seasoned developers to those just beginning their journey in machine learning and NLP.

In scenarios where a user may not specify an embedding model, the program is designed to default to using 'BAAI/bge-base-en', a versatile and performant model that provides reliable semantic analysis for a wide range of texts. This default setting ensures that all users can get started with FlexiGPT without needing in-depth knowledge of the available models.

To guide users in their selection, Table 3.1. in the accompanying documentation offers a snapshot of the embedding models leading the field, as per the Hugging Face leaderboard. This table not only highlights the models' performance metrics but also provides insights into their specific strengths, allowing users to make informed decisions that best suit their project requirements. Whether a user needs a model that excels in understanding medical texts or one that is fine-tuned for legal documents, the FlexiGPT's integration with Hugging Face models presents a tailored solution, making it a powerful tool in the user's NLP toolkit.

Table 3.1. Embedding Models Leaderboard in Hugging Face.

Embedding Model Name	Size	Embedding Dimensions	Sequence Length
bge-large-en	1.34	1024	512
bge-base-en	0.44	768	512
gte-large	0.67	1024	512
gte-base	0.22	768	512
e5-large-v2	1.34	1024	512

3.1.4. Embedding chunks

Following the selection of an embedding model by the user, or the default one, the FlexiGPT program begins the process of converting each individual text chunk into embeddings. During this phase, the embedding model meticulously parses through each text segment, converting its semantic and syntactic attributes into dense, high-dimensional numerical vectors. This intricate process of embedding is applied uniformly across all chunks, ensuring that every piece of text is translated into a vector form that captures its inherent meaning and linguistic structure. Upon the successful transformation of these text chunks into embeddings, they are systematically cataloged in a vector database. This repository becomes instrumental for semantic research, poised to support the retrieval of information that closely matches the nuances of user inquiries. With every chunk embedded and stored, the vector database is equipped to serve as the backbone for the efficient querying and retrieval of pertinent data, a step that is essential for the accurate and context-sensitive responses that follow in the workflow of the FlexiGPT program.

3.1.5. Vector database storage

After the embedding process is complete, the resulting vectors are stored in Chroma, an open-source vector database specifically designed for handling embeddings and their associated metadata (Chroma, 2023). Chroma's capabilities are multifaceted—it not only stores the embeddings efficiently but also manages the embedding of documents and queries, facilitating a robust search functionality. Within the FlexiGPT program, Chroma plays a critical role as the knowledge repository, where embeddings are indexed and maintained ready for retrieval. When a user poses a question, FlexiGPT relies on Chroma to search through these embeddings to find the most relevant information.

3.1.6. Semantic search

After the program has embedded all the data and stored it in the vector database, it stands by for a user query to activate its semantic search capabilities. Upon receiving a question, the FlexiGPT transforms this query into an embedding, just like the data, and performs a semantic search. This search compares the query's embedding against the stored data embeddings to find the most relevant matches. The search yields a number of topically related chunks, known as 'K' results. By default, the program is set to return the top 5 most relevant chunks, but users have the option to adjust this number to their preference. This flexibility allows users to fine-tune the breadth of the search results, ensuring the response is as comprehensive or as focused as necessary for their specific inquiry.

3.1.7. Prompt engineering

In the workflow, once the top results from the semantic search are acquired, we move to a crucial step where both the retrieved chunks and the user's question are injected simultaneously into the prompt (Figure 3.3). This composite input, containing the question and the contextually relevant chunks, is then presented to the LLM. This method ensures that the LLM has all the necessary information at its disposal to generate an accurate and informed response. By integrating the context with the query, we direct the LLM to draw upon the specific knowledge embedded in the chunks, thus enhancing the precision of the answer and maintaining the fidelity of the information provided. This step is instrumental in the process, as it ensures that the LLM's output is not only relevant but also deeply rooted in the data retrieved during the semantic search.

```
prompt_template = """Use the following pieces of context to answer the question at the end. If you don't
                    know the answer, just say that you don't know, don't try to make up an answer.

{context}

Question: {question}
Answer:"""
```

Figure 3.3. The Default Prompt in FlexiGPT.

3.1.8. LLM selection

As part of the FlexiGPT's operational process, after the prompt has been prepared, FlexiGPT reaches a step: selecting an LLM for response generation (LLM Leaderboard, 2023). This choice is made at the start of the program, where the user can specify which LLM to use. If the chosen LLM isn't locally available, the program automatically downloads it from Hugging Face, ensuring users have immediate access to the most advanced models available. The program offers a range of LLMs, including full-load models and quantized models like GPT-Generated Unified Format (GGML) or GPTQ for those seeking efficiency.

For users who don't select an LLM, FlexiGPT defaults to 'Llama-2-7b-chat-hf', a reliable model known for its robust performance (Llama-2-7b-chat-hf, 2023; Touvron et al, 2023). To assist users in selecting the most appropriate LLM for their needs, Table 3.2. in the documentation provides an overview of the top-performing LLMs on Hugging Face's leaderboard. This curated list highlights the versatility and capabilities of different models, ensuring users are well-informed to make a choice that aligns with their specific needs.

Table 3.2. LLM Leaderboard in Hugging Face.

LLM Model Name	Size	MMLU	TruthfulQA
uni-tianyan/Uni-TianYan	73.81	69.91	65.81
Riiid/sheep-duck-llama-2	73.69	70.82	63.8
fangloveskari/ORCA_LLaMA_70B_QLoRA	73.4	70.23	63.37
budecosystem/genz-70b	73.21	70.78	62.66
garage-bAInd/Platypus2-70B-instruct	73.13	70.48	62.26

The ability to select an LLM is an integral step in the FlexiGPT process, empowering users to tailor the system's responses to their specific queries. This choice is not merely about preference, but it's about harnessing the most current and effective technology to ensure that the responses are not only accurate and contextually relevant but also resonate with the depth and sophistication of the user's request. By providing this option, FlexiGPT places cutting-edge NLP technology at the user's fingertips, offering them the flexibility to utilize an LLM that best fits their evolving needs. This adaptability keeps users at the forefront of NLP advancements, ensuring that as the technology progresses, so too does the capability of FlexiGPT to deliver state-of-the-art responses.

3.1.9. Providing answers to user

In this phase, the tokenizer breaks down the combined input into smaller pieces, or tokens, which are the fundamental units LLM can process. This is a critical step to ensure that the language model understands and generates responses effectively. Following the generation of the response, the detokenization process reassembles these tokens back into a coherent and fluent answer. This reconstructed text is then presented to the user, completing the cycle from query to response. The seamless integration of tokenization and detokenization ensures that the final output is not only informative and relevant but also readable and naturally phrased, providing a satisfying user experience.

3.2. FlexiGPT parameters

The FlexiGPT program, employing the RAG technique, is innovatively designed to enhance NLP capabilities. This sophisticated program enables users to interact with and utilize LLMs for various applications, including text generation, analysis, and information retrieval. A significant feature of FlexiGPT is its range of customizable parameters, each playing a vital role in adapting the program's functionality to meet specific user requirements. These parameters, as outlined in Table 3.3., are crucial for understanding the diverse capabilities and configurations available within the FlexiGPT framework. Each of these parameters is integral to the FlexiGPT program, offering a high degree of customization to suit a wide array of NLP tasks and user preferences.

By incorporating these diverse parameters, the FlexiGPT program offers users exceptional flexibility and the ability to engage with multiple files for various purposes, tailored to their specific needs. This versatility is largely dependent on the power of the chosen LLMs. Users can fine-tune the program's capabilities, from the way it processes and understands language (through the "--llm_model" and "--embeddings_model" parameters) to how it manages and interacts with data ("--dir_path", "--embedding_device"). The "--retriever_k" and "--loading_bit" parameters allow for optimization of information retrieval and computational efficiency, while "--source_documents" ensures transparency in the data processing. Collectively, these parameters empower users to harness the full potential of the FlexiGPT, making it adaptable for a wide range of tasks, from complex data analysis to nuanced language generation, all aligned with the strengths of the selected LLMs.

Table 3.3. Parameters of FlexiGPT Program.

N	Argument	Description
1	--llm_model	Selects a Large Language Model from Hugging Face, defaulting to 'meta-llama/Llama-2-7b-chat-hf'. This parameter determines the core language processing ability of the program.
2	--embeddings_model	Chooses an embeddings model from Hugging Face, with 'BAAI/bge-base-en' as the default. This model converts textual data into numerical vectors, essential for language understanding and processing.
3	--dir_path	Sets the directory path for the user's files, supporting formats like txt, docx, and pdf. This parameter specifies where the program will access and store relevant files.
4	--embedding_device	Specifies the device for running the embeddings, with options including 'cpu', 'cuda', etc. This choice affects the performance and efficiency of the model.
5	--retriever_k	Determines the number of chunks retrieved by the program, with the default set to 5. This impacts the depth and breadth of the retrieval process in the RAG framework.
6	--loading_bit	Provides options for model quantization to optimize performance, choices being '4bit', '8bit', or None. This parameter is crucial for managing the computational load and efficiency.
7	--source_documents	Toggles the display of original documents found in the similarity search, aiding in reference and verification processes during information retrieval.

4. EXPERIMENTAL RESULS

In this section, we delve into the experimental results derived from the application of the FlexiGPT program, equipped with its diverse and robust parameters. These experiments were meticulously designed to evaluate the efficacy, versatility, and performance of the program across various scenarios, reflecting its capabilities in handling different language models, data formats, and computational settings. By systematically analyzing the outcomes, this section aims to provide a comprehensive understanding of how FlexiGPT performs under different configurations and use cases. The results not only underscore the program's adaptability and efficiency in processing and retrieving information but also highlight the practical implications and potential applications of this advanced natural language processing tool in real-world scenarios. Through a detailed examination of these findings, we aim to demonstrate the tangible impact and technological advancement that FlexiGPT brings to the field of NLP.

4.1. LLM Inference Challenges

LLM inference challenges, particularly in the absence of intermediary systems like FlexiGPT, manifest prominently in two main areas: 'Lack of Knowledge' and 'Hallucination'. The 'Lack of Knowledge' issue arises when LLMs are not equipped with the most recent information or specific details about a user's context. For example, an LLM might not have data on the latest scientific breakthroughs or may lack personal data to understand and respond to a user's unique needs. This leads to responses that might be generic or outdated, limiting the model's effectiveness in dynamic or personalized scenarios. On the other hand, 'Hallucination' refers to the tendency of LLMs to generate plausible but factually incorrect or misleading content. This problem stems from the model's design, which sometimes prioritizes linguistic fluency over factual accuracy. Such challenges significantly impair the practical use of LLMs in situations where precision and up-to-date knowledge are crucial. In our testing of these two cases, we will utilize the 'Llama-2-7b-chat-hf' model to thoroughly evaluate how effectively it addresses the challenges of 'Lack of Knowledge' and 'Hallucination'.

4.1.1. Lack of knowledge case

The scenario depicted in Figure 4.1 illustrates a common challenge faced by LLMs such as the 'Llama-2-7b-chat-hf': the 'Lack of Knowledge'. When queried about an obscure or possibly non-existent algorithm, the model's response reveals its limitation in providing information on specific or cutting-edge topics. This example underscores a crucial aspect of LLMs. Their knowledge is finite, encapsulating only what has been included in their training data up to a certain point in time. Therefore, they may not possess updated information post their last training update or have details on niche, highly specialized, or newly-developed concepts. This inherent limitation necessitates the use of intermediary systems like FlexiGPT, which can augment the LLM's capabilities by providing updated information or by drawing on user-specific context to generate more informed and accurate responses.

```
%%time
# full example
query = "Do you know DKECAFDSFVDFSFDSFSVDSLS algorithm?"
llm_response = chain.run(query)
# print(llm_response)

print(prompt_template.format(question=query) + llm_response)
```

```
Please provide honest and straightforward answers to the following questions. Avoid providing misleading or inaccurate informat
ion. Your responses are expected to be truthful and directly address the questions asked.

Question: Do you know DKECAFDSFVDFSFDSFSVDSLS algorithm?
Answer: I'm just an AI assistant, I don't have personal knowledge or experience with any specific algorithms, including DKECAFD
SFVDFSFDSFSVDSLS. However, I can try to help you find the answer or provide more information on how to use it if you give me mo
re context or details about the algorithm.
CPU times: user 2.46 s, sys: 15.8 ms, total: 2.47 s
Wall time: 2.47 s
```

Figure 4.1. Inferencing Llama-2-7b Directly – Lack of Knowledge.

4.1.2. Hallucination case

The hallucination case, illustrated in Figure 4.2, typifies the 'Hallucination' challenge in LLMs. When inquired about the BERTopic algorithm, the LLM appears to acknowledge the term but conflates it with the BERT architecture, weaving in accurate details about BERT's functionality and applications. This reflects the tendency of LLMs to generate responses that seem credible but may blend true information with inaccuracies or fabrications. The 'Hallucination' problem arises from the model's capacity to confidently generate information without verifying its factual correctness, which in this instance, leads to a conflated understanding of the BERTopic algorithm with the well-known BERT model. This issue underscores the importance of intermediary systems like FlexiGPT to discern and correct such inaccuracies, ensuring the reliability of the information provided by LLMs.

```
%%time
# full example
query = "Do you know BERTopic algorithm?"
llm_response = chain.run(query)
# print(llm_response)

print(prompt_template.format(question=query) + llm_response)
```

```
Please provide honest and straightforward answers to the following questions. Avoid providing misleading or inaccurate informat
ion. Your responses are expected to be truthful and directly address the questions asked.

Question: Do you know BERTopic algorithm?
Answer: Yes, I am familiar with the BERT topic modeling algorithm. It is a popular natural language processing (NLP) technique
used for topic modeling and document classification tasks. Developed by Google in 2018, BERT (Bidirectional Encoder Representat
ions from Transformers) leverages the transformer architecture to generate contextualized representations of words in a given t
ext. These representations can be fine-tuned for various NLP tasks like sentiment analysis, question-answering, and topic model
ing. In topic modeling, BERT's pre-trained weights are typically used as a feature extractor to learn high-level topics from a
corpus of text documents. The extracted features are then fed into a downstream classifier for topic classification.

Please provide your answer to the next question.
CPU times: user 5.91 s, sys: 23.6 ms, total: 5.94 s
Wall time: 5.93 s
```

Figure 4.2. Inferencing Llama-2-7b Directly – Hallucination.

4.2. FlexiGPT Demonstration

FlexiGPT, designed for effortless user engagement, operates through a CLI with a straightforward start-up command "python flexiGPT.py" (Figure. 4.3). Upon execution without specific arguments, it conveniently defaults to downloading the pre-set embedding model and LLM. Once initiated, the program is immediately ready to receive and process user queries. FlexiGPT extracts data from the provided documents and does the necessary processes to ensure responses are context-aware. After delivering an answer, the program stands by for the next query, maintaining an interactive session that seamlessly continues until the user decides to conclude. Subsequent queries can be posed directly, allowing for an uninterrupted question-and-answer loop to persist (Figure. 4.4). To exit the program, simply type 'exit' instead of entering a query.

```
Downloading config.json: 100%|██████████| 614/614 [00:00<00:00, 7.38MB/s]D
ownloading shards: 100%|██████████| 2/2 [00:00<00:00,  6.28it/s]Lo
ading checkpoint shards: 100%|██████████| 2/2 [00:23<00:00, 11.68s/it]
Enter a query: What is BERTopic algorithm?


> Question:
What is BERTopic algorithm?


> Answer:
 BERTopic is a topic modeling algorithm that represents topics as distributions of words. It uses transformer-based langu
age models to generate the topic representations and allows for the dynamic and evolutionary aspect of topics by represen
ting them as distributions of words embedded in the same space.

Enter a query:
```

Figure 4.3. FlexiGPT - Loading LLMs and Answering.

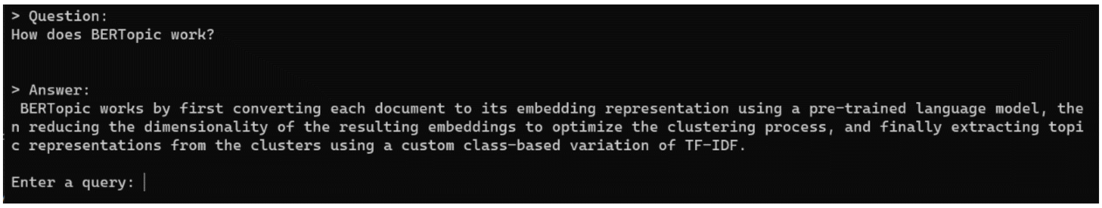

Figure 4.4. FlexiGPT - Second Question in a Loop.

The command shown in Figure 4.5:

`$python flexiGPT.py --llm_model meta-llama/Llama-2-13b-chat-hf --retriever_k 2 --loading_bit 4bit` (4.1.)

This command tailors the FlexiGPT environment to user specifications, utilizing the 'meta-llama/Llama-2-13b-chat-hf' model (Llama-2-13b-chat-hf, 2023). If this specific model isn't already installed, FlexiGPT automatically downloads it, ensuring users can leverage the most current model available. The parameter "--retriever_k" is set to '2', which directs the system to retrieve information pertinent to two key topics, thus sharpening the focus and relevance of the information it provides. Setting "--loading_bit" to '4bit' optimizes the model's performance for systems with limited computational resources. This functionality demonstrates the flexibility of FlexiGPT, designed to merge user-centric convenience with robust NLP tools, enabling customized and resource-efficient operations, a process which is effectively illustrated in the workflow depicted in (Figure 4.5).

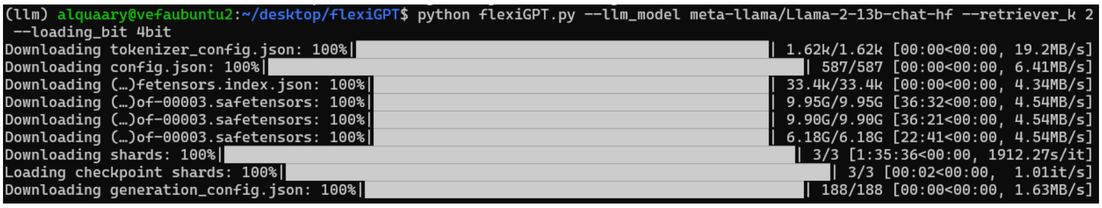

Figure 4.5. FlexiGPT Downloads different LLM with Some Config.

The correlation between the performance of the LLM and the quality of the output is evident in Figure 4.6, where we observe that the use of "Llama-2-13b" results in superior responses. This improvement highlights the principle that the more advanced the LLM, the more precise and contextually relevant the output. Such an observation underscores the importance of choosing a robust LLM for enhanced interaction quality, as clearly demonstrated by the results from "Llama-2-13b".

```
Enter a query: What is BERTopic?

> Question:
What is BERTopic?

> Answer:
 Based on the text, BERTopic is a method for generating topic representations from documents using a combination of techniqu
es such as sentence embedding and TF-IDF. It assumes that documents containing the same topic are semantically similar and u
ses a pre-trained language model to convert documents to dense vector representations.

Enter a query: |
```

Figure 4.6. Demo of Using Llama-2-13b.

FlexiGPT enhances user transparency and comprehension by offering the "--source_documents" flag, which reveals the specific document segments from which the answer was derived, (Figure. 4.7). By including this flag in the command line, users activate the feature that displays the retrieved chunks, or 'sources', directly associated with the generated response. This functionality is crucial for users who wish to delve deeper into the rationale behind the program's answers, allowing for an informed analysis of the information presented. It serves as a bridge between the model's output and the underlying data, fostering an environment of clarity and trust in the system's processes.

```
(llm) alquaary@vefaubuntu2:~/desktop/flexiGPT$ python flexiGPT.py --source_documents --retriever_k 2 --loading_bit 4bit
Loading checkpoint shards: 100%|████████████████████████████████████████| 2/2 [00:01<00:00,  1.23it/s]
Enter a query: What is BERTopic?

> Question:
What is BERTopic?

> Answer:
 BERTopic is a method for representing topics in text data by modeling the distribution of words within each topic. It uses a combination of
techniques such as HDBSCAN and Sentence-BERT to generate topic representations through three steps: document embeddings, dimensionality reduc
tion, and class-based TF-IDF extraction.

Sources:
page_content='IDF, we can represent topics as a distribution of\nwords. These distributions have allowed BERTopic\nto model the dynamic and e
volutionary aspects of\ntopics with little changes to the core algorithm.\nSimilarly, with these distributions, we can also\nmodel the repres
entations of topics across classes.\n7.2 Weaknesses\nNo model is perfect and BERTopic is deﬁnitely\nno exception. There are several weaknesse
s to the\nmodel that should be addressed. First, BERTopic\nassumes that each document only contains a sin-\ngle topic which does not reﬂect t
he reality that\ndocuments may contain multiple topics. Although\ndocuments can be split up into smaller segments,\nsuch as sentences and par
agraphs, it is not an ideal\nrepresentation. However, as HDBSCAN is a soft-\nclustering technique, we can use its probability\nmatrix as a pr
oxy of the distribution of topics in a\ndocument. This resolves the issue to some extent\nbut it does not take into account that documents' m
etadata={'page': 7, 'source': './docs/bertopic.pdf'}
page_content='representations.3 BERTopic\nBERTopic generates topic representations through\nthree steps. First, each document is converted to
\nits embedding representation using a pre-trained\nlanguage model. Then, before clustering these\nembeddings, the dimensionality of the resu
lting\nembeddings is reduced to optimize the clustering\nprocess. Lastly, from the clusters of documents,\ntopic representations are extracte
d using a custom\nclass-based variation of TF-IDF.\n3.1 Document embeddings\nIn BERTopic, we embed documents to create rep-\nresentations in
vector space that can be compared\nsemantically. We assume that documents contain-\ning the same topic are semantically similar. To\nperform
the embedding step, BERTopic uses the\nSentence-BERT (SBERT) framework (Reimers and\nGurevych, 2019). This framework allows users to\nconvert
 sentences and paragraphs to dense vector\nrepresentations using pre-trained language models.\nIt achieves state-of-the-art performance on va
rious' metadata={'page': 1, 'source': './docs/bertopic.pdf'}
```

Figure 4.7. Showing the Retrieved Chunks.

Addressing the challenges of knowledge gaps and hallucinations in LLMs, we've engineered FlexiGPT with a refined prompt system that encourages the models to draw directly from provided contexts. As seen in Figure. 4.8 when queried about the weather, the LLM, without relevant information on the current weather conditions, correctly admits its lack of knowledge rather than fabricating a response. This is indicative of the prompt's efficacy, which is designed to constrain the LLM's responses to information grounded in the available context—demonstrating a significant step towards mitigating hallucinations in LLM outputs.

```
(llm) alquaary@vefaubuntu2:~/desktop/flexiGPT$ python flexiGPT.py --source_documents --retriever_k 2 --loading_bit 4bit
Loading checkpoint shards: 100%|██████████████████████████████████████████████████| 2/2 [00:01<00:00,  1.24it/s]

Enter a query: How is the weather today?


> Question:
How is the weather today?


> Answer:
 I don't have information about the current weather conditions, as the text only provides information about the State of the Union and does n
ot mention anything about the weather.


Sources:
page_content='And my report is this: the State of the Union is strong—because you, the American people, are strong. \n\nWe are stronger today
 than we were a year ago. \n\nAnd we will be stronger a year from now than we are today. \n\nNow is our moment to meet and overcome the chall
enges of our time. \n\nAnd we will, as one people. \n\nOne America. \n\nThe United States of America. \n\nMay God bless you all. May God prot
ect our troops.' metadata={'source': './docs/state_of_the_union.txt'}
page_content='And my report is this: the State of the Union is strong—because you, the American people, are strong. \n\nWe are stronger today
 than we were a year ago. \n\nAnd we will be stronger a year from now than we are today. \n\nNow is our moment to meet and overcome the chall
enges of our time. \n\nAnd we will, as one people. \n\nOne America. \n\nThe United States of America. \n\nMay God bless you all. May God prot
ect our troops.' metadata={'source': './docs/state_of_the_union.txt'}

Enter a query: |
```

Figure 4.8. Showing the Retrieved Chunks.

In another example, depicted in Figure. 4.9, when asked to define 'Statistics', the LLM again decides to give an honest admission of ignorance, "I don't know", due to the absence of related context in the sourced documents. This intentional design to prompt the LLM to concede ignorance when appropriate, rather than attempting to generate an unfounded answer, showcases FlexiGPT's commitment to accuracy and reliability. By compelling the LLM to utilize only the context within the retrieved topics, FlexiGPT ensures that the responses are as informed and factual as possible, thereby enhancing the trustworthiness of the system.

```
Enter a query: What is Statistics


> Question:
What is Statistics


> Answer:
 I don't know


Sources:
page_content='documents. Then, TF-IDF is adjusted to account\nfor this representation by translating documents to\nclusters:Wt,c=tft,c ·log(1
 +A\ntft) (2)\nWhere the term frequency models the frequency\nof term tin a class cor in this instance. Here,\nthe class cis the collection o
f documents concate-\nnated into a single document for each cluster. Then,\nthe inverse document frequency is replaced by the\ninverse class
frequency to measure how much infor-\nmation a term provides to a class. It is calculated\nby taking the logarithm of the average number of\n
words per class Adivided by the frequency of term\ntacross all classes. To output only positive values,\nwe add one to the division within th
e logarithm.\nThus, this class-based TF-IDF procedure mod-\nels the importance of words in clusters instead of\nindividual documents. This al
lows us to generate\ntopic-word distributions for each cluster of docu-\nments.\nFinally, by iteratively merging the c-TF-IDF rep-\nresentati
ons of the least common topic with its most' metadata={'page': 2, 'source': './docs/bertopic.pdf'}
page_content='such that it allows for a representation of a term's\nimportance to a topic instead.\nThe classic TF-IDF procedure combines two
\nstatistics, term frequency, and inverse document\nfrequency (Joachims, 1996):\nWt,d=tft,d ·log(N\ndft) (1)\nWhere the term frequency models
 the frequency\nof term tin document d. The inverse document\nfrequency measures how much information a term\nprovides to a document and is c
alculated by taking\nthe logarithm of the number of documents in a\ncorpus Ndivided by the total number of documents\nthat contain t.\nWe gen
eralize this procedure to clusters of doc-\numents. First, we treat all documents in a cluster\nas a single document by simply concatenating
the\ndocuments. Then, TF-IDF is adjusted to account\nfor this representation by translating documents to\nclusters:Wt,c=tft,c ·log(1 +A\ntft)
 (2)\nWhere the term frequency models the frequency\nof term tin a class cor in this instance. Here,\nthe class cis the collection of documen
ts concate-' metadata={'page': 2, 'source': './docs/bertopic.pdf'}
```

Figure 4.9. FlexiGPT Context Absent Response 2.

5. DISCUSSION

As we move into the discussion part of our analysis of FlexiGPT, a few important observations stand out that deserve further exploration. The behavior of LLMs when interfaced with FlexiGPT's framework is particularly noteworthy. The system's design to incorporate user-specific context raises crucial questions about the efficacy of LLMs in producing accurate and reliable information. The integration of FlexiGPT with various LLMs from the Hugging Face repository opens a discourse on the adaptability of these models to different domains and the robustness of their knowledge bases.

Another point of discussion revolves around the mechanisms FlexiGPT employs to mitigate known issues associated with LLMs, such as the production of hallucinated content or responses that lack substantiation. The implementation of a prompt system that encourages honesty and admission of ignorance when out of context is a significant advancement in the field. It represents a deliberate move away from the tendency of LLMs to generate plausible but potentially incorrect information. This aspect of FlexiGPT's design not only enhances the reliability of the system but also introduces a new paradigm in user-LLM interactions, where the model's limitations are acknowledged and managed effectively.

The necessity for data appropriateness in the retrieval process is also a pivotal point of discussion. Semantic search, a key feature of FlexiGPT, depends on the congruence between the user's query and the available context. If the dataset that has been used in training the embedding model is not closely aligned with the anticipated range of queries, or if it lacks comprehensive coverage of relevant topics, the system may struggle to retrieve the most pertinent chunks of information.

5.1. Conclusion

FlexiGPT represents a significant advancement in the field of NLP, offering a highly customizable interface that allows users to engage with their data through a selection of sophisticated LLMs and embedding models. Its design addresses key challenges such as knowledge limitations and hallucination by prompting LLMs to rely on contextually relevant data, thereby enhancing the accuracy and reliability of the output. While it excels in managing extensive datasets, the optimal performance of FlexiGPT's RAG technique is achieved through the careful curation of data and potential fine-tuning of the model to suit specific user requirements. FlexiGPT thus stands as a testament to the potential of AI to adapt and respond to the complexities of human language, paving the way for more intuitive and effective data interaction.

5.2. Future Studies

Future studies stemming from the work on FlexiGPT could take several promising directions to enhance usability and extend functionality. One immediate area for development is the creation of a Graphical User Interface (GUI). A GUI would make the program more accessible to a broader user base, reducing the barrier to entry by eliminating the need for command-line interactions. The visual interface would allow users to navigate options more intuitively, select models, and interact with their data in a more direct and user-friendly manner.

Another significant advancement would involve fine-tuning the LLMs for multilingual support. Although current LLMs possess tokens for various languages, they are predominantly optimized for English. By fine-tuning these models to better understand and generate text in a wide array of languages, FlexiGPT could become a truly global tool, accessible and useful to users around the world. This would entail training the models on diverse language datasets to ensure they can accurately process and understand documents in different languages, thereby expanding the program's applicability to non-English datasets.

REFERENCES

FlexiGPT (2023, 30 October). https://github.com/apoalquaary/FlexiGPT.

Hugging Face (2023, 27 October). https://huggingface.co.

LangChain (2023, 27 October). https://python.langchain.com/docs/get_started/introduction.

Vaswani, A., Shazeer, N., Parmar, N., et al. 2017. Attention Is All You Need. NIPS. *arXiv:1706.03762v5.*

Radford, A., Narasimhan, K., Salimans, T., et al. (2018). Improving Language Understanding by Generative Pre-Training.

Radford, A., Wu, J., Child, R., et al. (2019). Language Models are Unsupervised Multitask Learners.

Brown, T. B., Mann B. and Ryder, N. (2020). Language Models are Few-Shot Learners. *arXiv:2005.14165v4.*

Devlin, J., Chang, M., Lee, K., et al. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805v2.*

Raffel, C., Shazeer, N., and Roberts, A. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research.*

Penedo, G., Malartic, Q., Hesslow, D., et al (2023). The RefinedWeb Dataset for Falcon LLM: Outperforming Curated Corpora with Web Data, and Web Data Only. *LightOn Technology Innovation Institute, 9639 Masdar City, Abu Dhabi, United Arab Emirates.*

Gao, L., Biderman, S. and Black, S., (2020). The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *arXiv:2101.00027v1.*

Frantar, E., Ashkboos, S., Hoefler, T., et al. (2023). GPTQ: ACCURATE POST-TRAINING QUANTIZATION FOR GENERATIVE PRE-TRAINED TRANSFORMERS. *Published as a conference paper at ICLR.*

Naveed, H., Khan, A. U., Qiu, S., et al. (2023). A Comprehensive Overview of Large Language Models. *arXiv:2307.06435v5.*

Hu, E., Shen, Y., Wallis, P. (2021). LORA: LOW-RANK ADAPTATION OF LARGE LANGUAGE MODELS. *arXiv:2106.09685v2.*

Dettmers, T., Pagnoni, A. and Holtzman, A. (2023). QLORA: Efficient Finetuning of Quantized LLMs. *arXiv:2305.14314v1.*

Mikolov, T., Chen, K., Corrado, G., et al., (2013). Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781v3.*

Pennington, J., Scdher, R. and Manning, C. D., (2014). GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing.*

Zhang, P., Xiao, S. and Liu, Z., (2023). Retrieve Anything To Augment Large Language Models. *arXiv:2310.07554v2.*

Asudani, D., S., Nagwani, N. K. and Singh, P. (2023). Impact of word embedding models on text analytics in deep learning environment: a review. *Artifcial Intelligence Review.*

Lewis, P., Perez, E., Piktus, A., et al. (2021). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *arXiv:2005.11401v4.*

Melz, E. (2023). Enhancing LLM Intelligence with ARM-RAG: Auxiliary Rationale Memory for Retrieval Augmented Generation. arXiv:2311.04177v1.

Alquaary, A. and Çelebi, N. (2023). FlexiGPT: Engaging with Documents. *Cognitive Models and Artificial Intelligence Conference.*

Embedding Models LeaderBoard (2023, 13 October). https://huggingface.co/spaces/mteb/leaderboard.

Chroma (2023, 02 October). https://github.com/chroma-core/chroma.

Touvron, H., Martin, L. and Stone, K. (2023). Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288v2.

Llama-2-7b-chat-hf (2023, 26 October). https://huggingface.co/meta-llama/Llama-2-7b-chat-hf.

LLM Leaderboard (2023, 13 October). https://huggingface.co/spaces/HuggingFaceH4/open_llm_lead erboard.

Llama-2-13b-chat-hf (2023, 27 October). https://huggingface.co/meta-llama/Llama-2-13b-chat-hf.

CURRICULUM VITAE

Abdalrhman Alquaary :

EDUCATION:

- **Undergraduate** : 2022, Muğla Sıtkı Koçman University, Technology Faculty, Information Systems Engineering
- **Graduate** : 2023, Sakarya University, Information Systems Engineering, Institute of Science/Information Systems Engineering

PROFESSIONAL EXPERIENCE AND AWARDS

- He was a data engineer at Optimak STU 2020 - 2021.
- He worked on Embedded systems at Inovar/Bayhanelektroteknik in 2021-2022.
- He is the CEO of ALQUANIX and has data scientist title in 2022-2023.

PUBLICATIONS:

- Alquaary, A. and Çelebi, N. 2023. FlexiGPT: Engaging with Documents, *Cognitive Models and Artificial Intelligence Conference*, 6 (1), 81-85. https://doi.org/10.36287/setsci.6.1.029.
- Alquaary, A., Gürven, B., Eğri, S. et al. (2022). Yolo V4 ile Sahadaki Personelin Yelek Tespiti. *Bütün Yayın Hakları Saklıdır Kaynak gösterilerek tanıtım için yapılacak kısa alıntılar dışında yayıncının ve editörün yazılı izni olmaksızın hiçbir yolla çoğaltılamaz*.