Building Customized Chatbots for Document Summarization and Question Answering using Large Language Models using a Framework with OpenAI, Lang chain, and Streamlit

Sangita Pokhrel[1], Swathi Ganesan[2], Tasnim Akther[3], Lakmali Karunarathne[4]

[1,2,4]Computer Science and Data Science, York St John University, London, United Kingdom

[3]Department of Library and Learning Services, York St John University, London, United Kingdom

E-mail: [1]s.pokhrel@yorksj.ac.uk, [2]s.ganesan@yorksj.ac.uk, [3]t.akther@yorksj.ac.uk, [4]l.karunarathne@yorksj.ac.uk

Abstract

This research presents a comprehensive framework for building customized chatbots empowered by large language models (LLMs) to summarize documents and answer user questions. Leveraging technologies such as OpenAI, LangChain, and Streamlit, the framework enables users to combat information overload by efficiently extracting insights from lengthy documents. This study discussed the framework's architecture, implementation, and practical applications, emphasizing its role in enhancing productivity and facilitating information retrieval. Through a step-by-step guide, this research has demonstrated how developers can utilize the framework to create end-to-end document summarization and question-answering applications.

DOI: https://doi.org/10.36548/jitdw.2024.1.006
Received: 04.02.2024. received in revised form: 06.03.2024, accepted: 22.03.2024, published: 07.04.2024

Keywords: Langchain, PDF Summarizer, Streamlit, OpenAI APIs, ChatBots, Large Language Models

1. Introduction

In an era typified by the exponential growth of digital information, the capacity to rapidly extract important insights from huge amounts of text has become paramount. Chatbots, developed using artificial intelligence (AI) and natural language processing (NLP) technology, have emerged as adaptable solutions capable of tackling this difficulty. With applications ranging from customer service to educational help, chatbots offer various paths for automating tasks such as document summarizing, and question answering, hence boosting productivity and knowledge retrieval [1].

Large language models (LLMs) represent the main element of recent breakthroughs in NLP, as they transform the way machines interpret and generate human-like text. These models, trained on vast datasets, demonstrate exceptional capabilities in comprehending and creating natural language, making them perfect candidates for powering specialized chatbots to document summarization and question-answering tasks [2]. The value of LLMs in NLP tasks cannot be emphasized since they enable chatbots to analyze and synthesize complicated textual information with unparalleled accuracy and efficiency [3]. The rationale for designing customized chatbots for document summarizing and question-answering originates from the inherent challenges caused by information overload. As the volume of digital information continues to rise dramatically, individuals and organizations are flooded with massive amounts of textual data, making it increasingly difficult to extract meaningful insights promptly. Customized chatbots equipped with document summarizing and question-answering capabilities offer a solution to this difficulty, enabling users to effectively explore and derive meaningful insights from vast quantities of text [4].

OpenAI's GPT models, in particular, have gained notable attention in the field of NLP because of their capacity to provide logical and contextually relevant text. Exclusively pre-trained on vast amounts of text data, these models stand as the backbone of this platform , providing the core language understanding capabilities necessary for document summarizing, and question-answering activities [5].On the other hand, LangChain enhances the capabilities of OpenAI's GPT models by offering a framework for performing linguistic processing tasks efficiently. With its modular architecture and wide support for diverse NLP activities, LangChain supports the integration of language models into chatbot applications effortlessly [6]. Streamlit serves as the user interface foundation for our proposed framework, enabling developers to design intuitive and interactive interfaces for chatbots. With its simplicity and versatility, Streamlit has improved the creation and deployment of chatbot apps, allowing users to interact seamlessly with the system [7].

In this research, a comprehensive architecture for constructing customized chatbots empowered by LLMs to handle the aforementioned difficulties has been presented. Leveraging state-of-the-art technologies such as OpenAI's GPT (Generative Pretrained Transformer) models, LangChain, and Streamlit, this framework offers a versatile solution for developers and researchers seeking to harness the power of LLMs for document summarization and question-answering tasks. By integrating these frameworks, this technique supports the seamless construction and deployment of chatbots capable of parsing, summarizing, and answering queries based on textual input.

2. Background Study

In this research review, the progress of NLP technology and the essential role of LLMs in redefining the capabilities of chatbots has been examined along with the importance of personalized chatbots that can summarise documents and answer questions to address the difficulties caused by an excessive amount of information. The article [1] utilized extractive summarizers to refine the main components of research papers, aiming to address information overload in the scientific literature. Through experiments, they found that enhancements considering the text's sophisticated structure improved summarization, especially for languages lacking refined NLP tools. This approach helped overcome challenges in extracting relevant insights from vast amounts of scientific text, emphasizing the need for deep text understanding for high-quality summaries. [2] explored conversational interfaces, specifically chatbots, in the context of AI ethics. The study explored the differences between scenario-based and large language model (LLM)-based chatbots in generating recommendations and discussed the ethical implications of LLM-based recommendations. By comparing their characteristics and limitations, the study highlighted concerns such as transparency, fairness, privacy, and accountability. This study concluded by stressing the importance of ethical considerations in developing and evaluating conversational AI systems, urging further research and industry efforts to implement ethical practices.

The researcher [3] employed machine learning to develop an evolving connectionist text summarizer, aiming to address challenges in automatic text summarization. By acknowledging the dynamic nature of linguistic attributes, the authors found that adaptive structures in connectionist architectures enhanced summarization effectiveness. This approach overcame limitations in modeling evolving systems, emphasizing the importance of language-independent strategies for accurate text summarization.

The rapid evolution of automatic text summarization technology in response to the exponential growth of online content has been investigated in [4]. It emphasized the importance of these summarizers in efficiently managing vast amounts of information by condensing it into concise yet comprehensive summaries while preserving the original meaning. This study explored both extractive and abstractive text summarization methods, highlighting the role of natural language processing (NLP) techniques and modern approaches such as Google's PEGASUS model. A detailed analysis of the proposed PEGASUS method was presented, and its architecture and sequence-to-sequence learning approach

Re discussed. The results showcased the model's ability to generate accurate and human-like summaries across various dimensions. The paper concludes by emphasizing the significance of automatic text summarization tools in meeting the increasing demand for accessible and user-friendly solutions, especially in the mobile domain, and proposes future directions for their development to cater to a wider audience. The authors of [8] developed a content generation tool utilizing the OpenAI language model, specifically GPT-3, as an API to streamline content creation for businesses and individuals. Powered by advanced machine learning algorithms, including a recurrent neural network (RNN) architecture, the tool aimed to efficiently produce high-quality content across various platforms. With a user-friendly dashboard, it offered a range of features, such as Facebook ads, LinkedIn posts, Amazon product descriptions, and blogs. Addressing the challenges of limited writing skills and time constraints, the tool emphasized its efficacy in simplifying content creation across multiple platforms, highlighting its value in assisting users with diverse content generation needs.

The study of [9] introduced LangChain, a query system utilizing LLMs for efficient information retrieval from PDF documents. By employing natural language processing algorithms and Streamlit, the authors found that LangChain streamlined the querying process and enhanced information retrieval. This approach overcame challenges in extracting relevant information from PDFs, offering a valuable tool for efficient data access. The authors of [10] explored a new approach to text summarization models by considering large language models (LLMs) like GPT-3.5 as reference or gold-standard oracles. They investigated the implications of using LLMs as references for model training and evaluation practices. The research investigated two LLM-based methods for evaluating summary quality, namely GPTScore and GPTRank, in conjunction with contrastive learning training techniques leveraging LLM-guided signals. Experiments conducted on CNN/DailyMail and XSum datasets demonstrated that smaller summarization models could attain comparable performance to LLMs when evaluated using LLM-based criteria.

Nonetheless, human evaluation revealed a disparity, suggesting that despite enhancements from the proposed training methods, smaller models have not yet reached the performance level of LLMs. The research highlighted the risks as well as the benefits of the LLM-as-reference setting, emphasizing the importance of further examination and improvement. It contributed by demonstrating empirical improvements in smaller models trained with LLM references and contrastive learning, while also revealing limitations in LLM-based training and evaluation methods.

In their study outlined in reference [6], researchers explored the utilization of LLMs for rapid application development, centering on LangChain, an accessible open-source software library.Highlighting LLMs like OpenAI's ChatGPT, renowned for tasks such as essay writing and code generation, the research underscored LangChain's modular structure. By showcasing practical examples across autonomous agents, chatbots, and document-based question answering, the study illuminated LangChain's prowess in accelerating application development. It emphasized the revolutionary impact of LLMs in the AI landscape and positioned LangChain as a pivotal tool in streamlining the development process, fostering ongoing exploration and innovation in the domain. The research of [11] investigated the problem-solving capabilities of LLMs like LaMDA, OPT, and GPT-3 in math word problems. Using the SVAMP dataset, GPT-3's “davinci-002” model showed robust performance on both symbolic and numeric problems, with a two-step approach improving accuracy in the numerical test set. Specific prompting techniques enhanced the model's ability to explain its thought process and solve complex problems. The study suggested that large LLMs can effectively solve symbolic math problems but highlights room for improvement.

Researchers of [12] by integrating an external knowledge management module aimed to enhance the Large Language Models (LLMs) like ChatGPT, enabling access to data from vector databases and the Internet. Focused on raising awareness about blockchain technology in Kazakhstan, a chatbot prototype was developed using the double diamond design process, incorporating frontend and backend layers with components like Prompt Templates, Chains, Memory, Models, and Agents. Python and OpenAI's GPT-3.5 were chosen for their versatility in natural language processing. Semantic and web search integration improved the chatbot's ability to provide real-time, accurate information. The study concluded that the developed chatbot system holds promise in improving citizen awareness, with future research aiming to adapt the architecture for the Kazakh language.

In the paper [13], they have introduced an approach for abstractive text summarization applying deep learning, aiming for precise and coherent summaries without redundancy. It employed an Encoder-decoder architecture with Bi-LSTM and attention mechanisms to reduce repetition as well as to enhance the contextual phrase generation. The model focused on multi-sentence summarization using the Double Attention Pointer Network and employed data processing methods such as cleansing, padding, and tokenizing. The proposed model, utilizing T5 for conditional generation and evaluated using ROUGE scores, shows competitive performance compared to the SASSBOOK TOOL with Daily Mail dataset and CNN , achieving a ROUGE score of 71%. The paper concluded that the proposed algorithm yields promising outcomes, outperforming existing methods in terms of semantic and syntactic structure, and suggests future work to enhance the system by including the paraphrasing method that has a abstractive summarizer.

Article [14] presented extractive text summarization as a solution to data overload across various domains. It investigated the use of Elmo embedding for generating summaries efficiently. Preprocessing involved text splitting, punctuation removal, tokenizing, stop word removal, and lemmatizing. Elmo embedding converted text into vectors, capturing contextdependent aspects. Sentences were scored using cosine similarity, and the top five formed the summary. Elmo embedding facilitated accurate summary generation. Future work includes improving summarization for multiple documents and enhancing speed and accuracy.

[15] The proposal suggests integrating LLMs, including GPT-based technologies, into multiagent systems (MASs) to improve the communication and decision-making capabilities. A novel agent architecture, that is based on the MAPE-K model, has been devised to enhance conversational features, reasoning, and decision-making abilities. Through a marketplace scenario, integration showcased the potential to revolutionize agent interactions and problem-solving capabilities within MASs. Challenges such as computational overhead and interpretability of decisions were acknowledged for future improvement. [16] created a web application that aimed to create concise summaries of YouTube video transcripts using NLP techniques and the Flask framework to overcome challenges in accessing video content efficiently. By providing features such as translation and text-to-speech, the authors found that the summarizer improved user accessibility to video content, overcoming challenges in understanding lengthy videos.

In reference [17], an innovative system has been introduced to address limitations in current technologies for building large-scale model applications. This system integrates the LangChain framework, enabling seamless connectivity between AI models like ChatGLM-6B and local data sources. Additionally, the system incorporated the Rasa framework, which enhanced the ability to perform tasks such as intent classification and entity recognition. By leveraging these frameworks, the system improved text generation efficiency, enabling more effective applications in various contexts, such as cloud network monitoring and scheduling systems.

From the literature review, it is evident that the integration of AI and NLP technologies, particularly through large language models (LLMs), offers promising solutions for addressing information overload. Chatbots powered by these technologies excel in tasks such as document summarization and question answering, demonstrating their efficiency in managing vast amounts of digital information across various domains.

3. Framework Architecture

OpenAI is a research organization dedicated to advancing artificial intelligence technologies for the betterment of society that conducts research across diverse domains in AI, including NLP, robotics, reinforcement learning, and beyond. A primary objective is to create AI systems capable of executing a diverse array of tasks with human-like intelligence. They've engineered AI systems capable of significant projects, including the creation of large-scale language models like the GPT (Generative Pre-Trained Transformer) series, which can generate text resembling human language based on input data. These models have applications in natural language comprehension, text generation, translation, etc. The framework of the system is to develop a web application that summarizes the pdf using the Streamlit, Langchain, and OpenAI APIs in which the architecture of the model can be seen in figure 1.

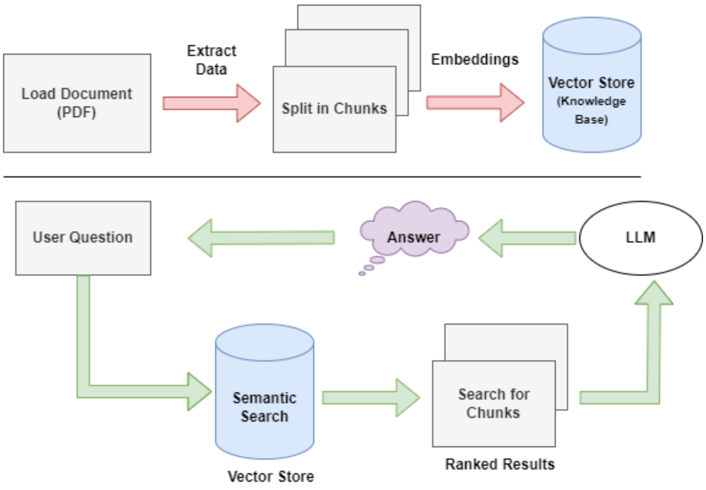


Figure 1. Architecture of the Model

Initially, the PDF document is uploaded into the system, followed by the importation of the PdfReader class from the PyPDF2 module. This Python library facilitates working with PDF files, allowing tasks like reading, manipulation, and text extraction from PDF documents.

The whole document will be split into chunks, creating an embedding from each chunk, and all the chunks will be stored in the vector database. In the context of data processing, a chunk is the segment of the data that has been divided from a larger dataset. It is often employed to break down into more manageable units for processing, analysis, or storage. Embeddings are quantitative representations of data that encapsulate its semantic meaning. In natural language processing (NLP), word embeddings are frequently employed to depict words as vectors within a high-dimensional space. This representation arranges words with akin meanings closer together in the vector space. These embeddings can be generated through diverse techniques, including Word2Vec, GloVe, or deep learning models like transformers. In this paper, we utilized the OpenAI embeddings class from the LangChain embeddings. OpenAI module. It is a part of the LangChain package and provides embeddings based on OpenAI's language models for text data. A vector store, also known as a knowledge base, is a repository where embeddings or vectors representing data are stored. The embeddings generated from the extracted data are stored in a vector store, which acts as a repository of knowledge that the system can reference during the search process. This vector store facilitates semantic search and retrieval of relevant information based on user queries.

When the user asks a question i.e prompt, it performs a semantic search in the vector store and searches for the chunks to find the ranked results. Based on the large language model, the OpenAI API has been integrated at the backend, which answers the user. If the user is satisfied with the answer, they can use it; if not, the user will ask with a more detailed customized prompt and wait for the specific answer. The figure 2 defines the proposed system block diagram.

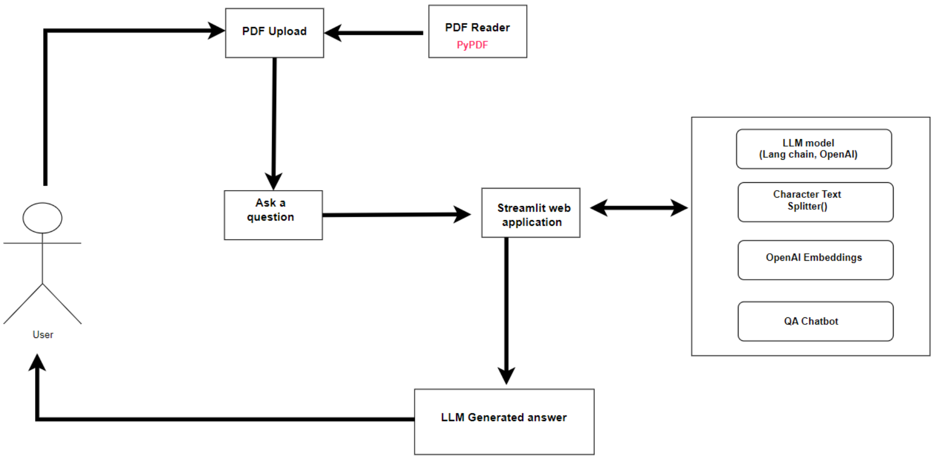
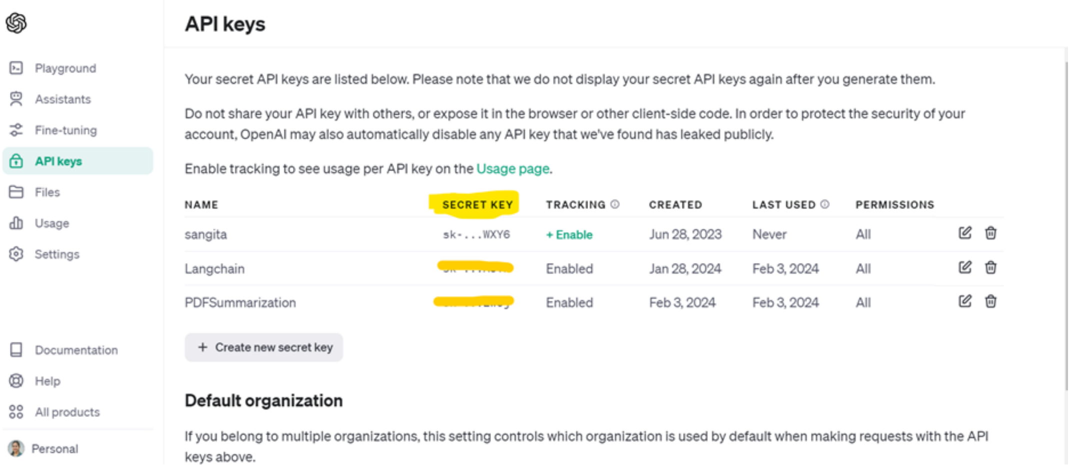


Figure 2. Proposed Block Diagram

The OpenAI API serves as a gateway to cutting-edge language processing capabilities, enabling access to advanced LLMs such as GPT-3, GPT-3.5, and GPT-4. These models excel in generating high-quality text across various styles and tones, simplifying tasks such as content creation, summarization, and script generation. LangChain, an open-source library, complements the OpenAI API by seamlessly integrating with other natural language processing (NLP) tools, facilitating the creation of robust pipelines for tasks such as data cleaning and summarization. Streamlit further enhances this ecosystem by simplifying the development of interactive web applications, allowing users to present outputs generated by OpenAI and LangChain with minimal coding requirements. Together, these tools empower users to leverage advanced language processing capabilities, streamline workflows, and efficiently create professional-looking applications.

4. Results and Implementation

The open AI API secret key was created from the OpenAI website and saved in an environment that needs to be treated like the password and avoid sharing it publicly. Once the secret key has been created (shown in Figure 3), it can be used for accessing the OpenAI API and integrate it into our application, projects, and research.



API keys

Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically disable any API key that we've found has leaked publicly.

Enable tracking to see usage per API key on the Usage page.

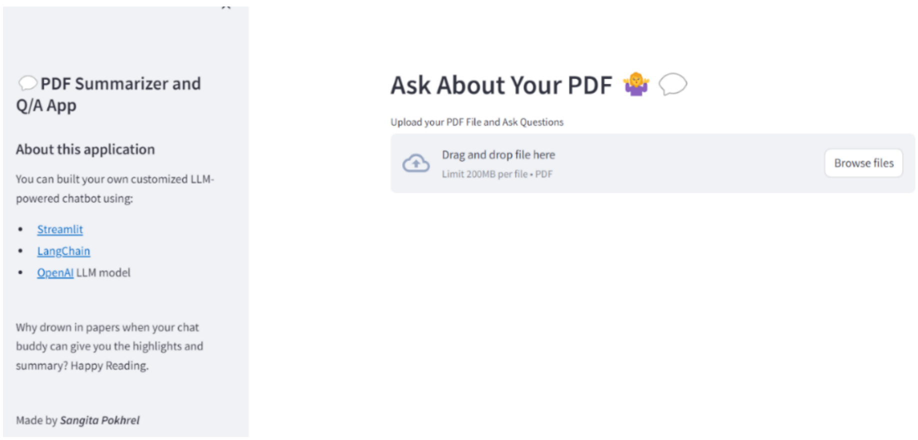
NAME	SECRET KEY	TRACKING	CREATED	LAST USED	PERMISSIONS
sangita	sk-...WXY6	+ Enable	Jun 28, 2023	Never	All
Langchain	[redacted]	Enabled	Jan 28, 2024	Feb 3, 2024	All
PDFSummarization	[redacted]	Enabled	Feb 3, 2024	Feb 3, 2024	All

Default organization

If you belong to multiple organizations, this setting controls which organization is used by default when making requests with the API keys above.

Figure 3. Creating the OpenAI API Key

The web application was designed using Streamlit, which is a promising open-source Python library that enables developers to build interfaces most easily. It simplifies the process of creating interactive and customizable web-based user interfaces (UIs) directly from Python scripts without needing knowledge of web development languages such as HTML, CSS, or JavaScript. Users can create interactive components such as sliders, buttons, text inputs, and data visualizations with only a few lines of code [18].



💬 PDF Summarizer and Q/A App

About this application

You can built your own customized LLM-powered chatbot using:

- Streamlit
- LangChain
- OpenAI LLM model

Why drown in papers when your chat buddy can give you the highlights and summary? Happy Reading.

Made by *Sangita Pokhrel*

Ask About Your PDF 🤷‍♀️ 💬

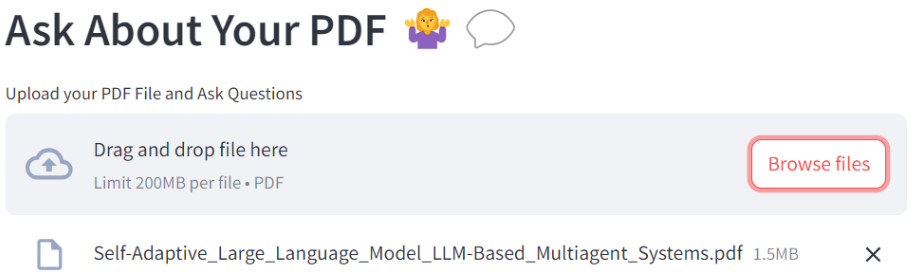
Upload your PDF File and Ask Questions

Drag and drop file here

Limit 200MB per file • PDF

Figure 4. Interface of the Application

Figure 4 shows the interface of our web application, where the information has been kept on the left side of the web and the ask about your pdf has been the main target where it takes pdf files. This whole system has been integrated with Streamlit, long-chain, and OpenAI large language models. When people are trying to save time and quickly want to receive a response from that pdf, they can simply ask the chatbot any questions, and answers can be obtained by searching in the pdf.



Ask About Your PDF 🤷‍♀️ 💬

Upload your PDF File and Ask Questions

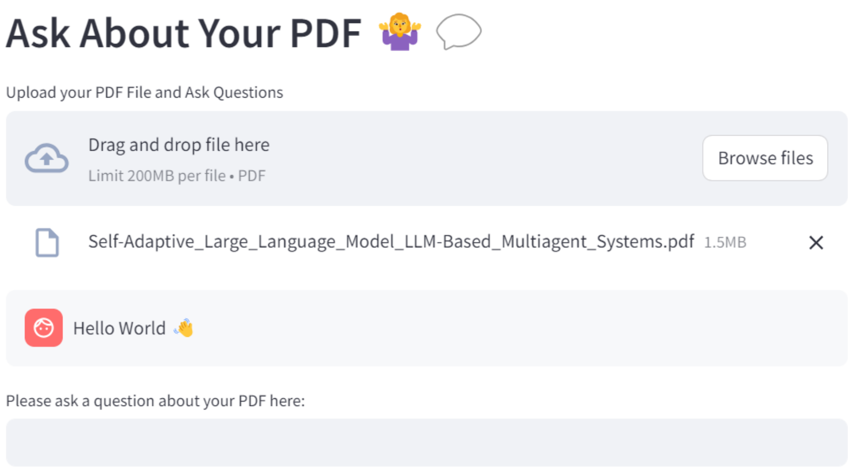
Drag and drop file here

Limit 200MB per file • PDF

Self-Adaptive_Large_Language_Model_LLM-Based_Multiagent_Systems.pdf 1.5MB ×

Figure 5. Uploading the pdf

Initially, the script verifies whether a PDF file has been uploaded (i.e., checks if the variable ‘pdf’ is None). Upon confirmation of an uploaded PDF file, it instantiates a PdfReader object to access the contents of the PDF (shown in figure 5). It iterates through PDF document to retrieve the text from each page. Finally, it concatenates the extracted text from all pages into a unified string variable.


Ask About Your PDF
Upload your PDF File and Ask Questions
Drag and drop file here
Limit 200MB per file • PDF
Browse files
Please ask a question about your PDF here:

Self-Adaptive_Large_Language_Model_LLM-Based_Multiagent_Systems.pdf 1.5MB

Hello World 👋

Figure 6. Q/A Chatbot for Question-Answering

Once the PDF is uploaded, the question-answering chatbot is initialized and a question from the user about the PDF is asked (as shown in figure 6).

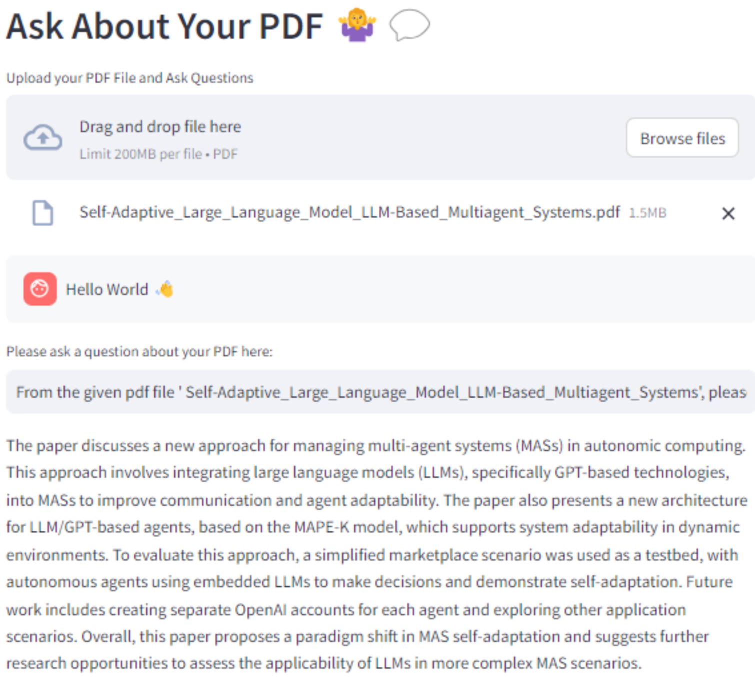
The prompt given by the user is:

Prompt 1

From the given pdf file 'Self-Adaptive_Large_Language_Model_LLM-Based_Multiagent_Systems', please summarize the pdf in approximately 200 words including all the details that have been discussed in the paper (shown in figure 7).

Results

This paper presents a novel method for overseeing multiagent systems (MASs) within autonomic computing. This method entails incorporating large language models (LLMs), particularly GPT-based technologies, into MASs to enhance communication and the adaptability of agents. Additionally, this paper introduces a fresh architecture for LLM/GPT-based agents founded in the MAPE-K model, fostering system adaptability within dynamic environments. To assess this approach, a simplified marketplace scenario served as a testbed, where autonomous agents utilized embedded LLMs to make decisions and exhibit self-adaptation. The future work will include establishing individual OpenAI accounts for each agent and investigating alternative application scenarios. overall, this paper proposes for a paradigm shift in MAS self-adaptation and indicates further research prospects to evaluate the viability of LLMs in increasingly intricate MAS scenarios.


Ask About Your PDF 🤷 💬
Upload your PDF File and Ask Questions
Drag and drop file here
Limit 200MB per file • PDF
Browse files
Self-Adaptive_Large_Language_Model_LLM-Based_Multiagent_Systems.pdf 1.5MB
Hello World 👋
Please ask a question about your PDF here:

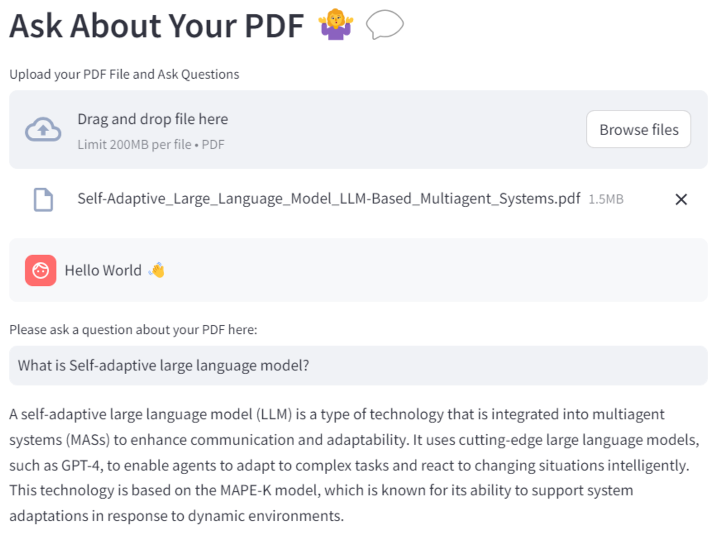
From the given pdf file ' Self-Adaptive_Large_Language_Model_LLM-Based_Multiagent_Systems', pleas

The paper discusses a new approach for managing multi-agent systems (MASs) in autonomic computing. This approach involves integrating large language models (LLMs), specifically GPT-based technologies, into MASs to improve communication and agent adaptability. The paper also presents a new architecture for LLM/GPT-based agents, based on the MAPE-K model, which supports system adaptability in dynamic environments. To evaluate this approach, a simplified marketplace scenario was used as a testbed, with autonomous agents using embedded LLMs to make decisions and demonstrate self-adaptation. Future work includes creating separate OpenAI accounts for each agent and exploring other application scenarios. Overall, this paper proposes a paradigm shift in MAS self-adaptation and suggests further research opportunities to assess the applicability of LLMs in more complex MAS scenarios.

Figure 7. Responses Generated by the Model

Prompt 2: What is the Self-adaptive Large Language Model?

Result: A self-adaptive large language model (LLM) is a type of technology that is integrated into multiagent systems (MASs) to enhance communication and adaptability. It uses cutting-edge large language models, such as GPT-4, to enable agents to adapt to complex tasks and react intelligently to changing situations. This technology is based on the MAPE-K model, which is known for its ability to support system adaptations in response to dynamic environments (shown in figure 8).


Ask About Your PDF 🤷‍♀️ 💬
Upload your PDF File and Ask Questions
Drag and drop file here
Limit 200MB per file • PDF
Browse files
Self-Adaptive_Large_Language_Model_LLM-Based_Multiagent_Systems.pdf 1.5MB
Hello World 👋
Please ask a question about your PDF here:

What is Self-adaptive large language model?

A self-adaptive large language model (LLM) is a type of technology that is integrated into multiagent systems (MASs) to enhance communication and adaptability. It uses cutting-edge large language models, such as GPT-4, to enable agents to adapt to complex tasks and react to changing situations intelligently. This technology is based on the MAPE-K model, which is known for its ability to support system adaptations in response to dynamic environments.

Figure 8. Results for Prompt 2

The original paper described a self-adaptive LLM-based multiagent system [15] in which researchers proposed the development of a novel LLM/GPT-based agent architecture along with the reasoning and decision-making capacities of these models.

Our experimental evaluation involved diverse documents covering various domains, where this system demonstrated significant effectiveness on pdf summary. This system efficiently got the key information from documents into concise summaries, surpassing traditional methods. The question-answering chatbot exhibited rapid responses within a few seconds per query with great user satisfaction. It first fetches documents from a retriever and subsequently employs a question-answering chain to respond to queries based on the retrieved documents. Specifically, this system utilized the ‘stuff’ document chain type, wherein a list of documents is incorporated into a prompt, which is then processed by a Large Language Model (LLM). Semantic search enhances search engine capabilities by understanding the intent and context of search queries, leading to more accurate and relevant search results.

5. Limitations and Discussions

The framework has some limitations where it relies on pre-trained language models, like OpenAI’s GPT models, which may not always understand specialized topics well or summarize complex documents accurately. While these models excel in generating human-like text, their summarization performance can be hindered by documents outside their training data scope. Additionally, dependence on cloud-based APIs like OpenAI raises concerns regarding data privacy, security, and long-term service availability. Hence, careful consideration and potential fine-tuning are essential to ensure the chatbot's robustness and reliability, particularly in diverse and specialized domains.

6. Conclusion and Future Recommendations

In conclusion, this research concludes with an in-depth approach to developing personalized chatbots derived from large language models (LLMs), emphasizing question response and document summarization by integrating technologies such as Streamlit, LangChain, and OpenAI, the framework effectively addresses the issue of information overload by facilitating the extraction of insights from documents. This research shows how developers can use the framework to build end-to-end applications for question-answering and document summarization by offering a step-by-step tutorial. The integration of OpenAI's advanced language models, LangChain's efficient NLP processing, and Streamlit's user-friendly interface design offer a versatile solution for researchers and developers seeking to harness the power of LLMs for text-based tasks.

Future recommendations include fine-tuning the model, integrating adaptive generative AI models, and expanding the capabilities of customized chatbots with a wider range of features. This framework has the potential to revolutionize the way users interact with and derive insights from textual data, enhancing productivity and facilitating knowledge retrieval across various domains.

References

[1] Balage Filho, Pedro Paulo, TA Salgueiro Pardo, and M. das Gracas Volpe Nunes. "Summarizing scientific texts: Experiments with extractive summarizers." In Seventh International Conference on Intelligent Systems Design and Applications (ISDA 2007), pp. 520-524. IEEE, 2007.

[2] Bang, Junseong, Byung-Tak Lee, and Pangun Park. "Examination of Ethical Principles for LLM-Based Recommendations in Conversational AI." In 2023 International Conference on Platform Technology and Service (PlatCon), pp. 109-113. IEEE, 2023.

[3] Prasad, Rajesh S., U. V. Kulkarni, and Jayashree R. Prasad. "Machine learning in evolving connectionist text summarizer." In 2009 3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication, pp. 539-543. IEEE, 2009.

[4] Nalini, N., Agrim Narayan, Akshay Mambakkam Sridharan, and Arkon Pradhan. "Automated Text Summarizer Using Google Pegasus." In 2023 International Conference on Smart Systems for applications in Electrical Sciences (ICSSES), pp. 1-4. IEEE, 2023.

[5] Patil, Dinesh D., Dhanraj R. Dhotre, Gopal S. Gawande, Dipali S. Mate, Mayura V. Shelke, and Tejaswini S. Bhoye. "Transformative trends in generative ai: Harnessing large language models for natural language understanding and generation." International Journal of Intelligent Systems and Applications in Engineering 12, no. 4s (2024): 309-319.

[6] Topsakal, Oguzhan, and Tahir Cetin Akinci. "Creating large language model applications utilizing langchain: A primer on developing llm apps fast." In International Conference on Applied Engineering and Natural Sciences, vol. 1, no. 1, pp. 1050-1056. 2023.

[7] Monks, Thomas, and Alison Harper. "Improving the usability of open health service delivery simulation models using Python and web apps." NIHR Open Research 3 (2023).

[8] Pokhrel, Sangita, and Shiv Raj Banjade. "AI Content Generation Technology based on Open AI Language Model." Journal of Artificial Intelligence and Capsule Networks 5, no. 4 (2023): 534-548.

[9] S, Adith Sreeram A, and Pappuri Jithendra Sai. “An Effective Query System Using LLMS and Langchain.” International Journal of Engineering Research & Technology, July 4, 12(6), 2023. 367 -369

[10] Liu, Yixin, Alexander R. Fabbri, Pengfei Liu, Dragomir Radev, and Arman Cohan. "On learning to summarize with large language models as references." arXiv preprint arXiv:2305.14239 (2023).

[11] Gaur, Vedant, and Nikunj Saunshi. "Symbolic math reasoning with language models." In 2022 IEEE MIT Undergraduate Research Technology Conference (URTC), pp. 1-5. IEEE, 2022.

[12] Mansurova, Aigerim, Aliya Nugumanova, and Zhansaya Makhambetova. "Development of a question-answering chatbot for blockchain domain." Scientific Journal of Astana IT University (2023): 27-40.

[13] Shibi, Krithick, R. Kingsy Grace, and M. Sri Geetha. "Abstractive Summarizer using Bi-LSTM." In 2022 International Conference on Edge Computing and Applications (ICECAA), pp. 1605-1609. IEEE, 2022.

[14] Gupta, Hritvik, and Mayank Patel. "Study of extractive text summarizer using the elmo embedding." In 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), pp. 829-834. IEEE, 2020.

[15] Nascimento, Nathalia, Paulo Alencar, and Donald Cowan. "Self-adaptive large language model (llm)-based multiagent systems." In 2023 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C), pp. 104-109. IEEE, 2023.

[16] Kumar, Jitender, Ritu Vashistha, Roop Lal, and Dhrumil Somanir. "YouTube Transcript Summarizer." In 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), pp. 1-4. IEEE, 2023.

[17] Duan, Zhihua. "Application development exploration and practice based on LangChain+ ChatGLM+ Rasa." In 2023 2nd International Conference on Cloud Computing, Big Data Application and Software Engineering (CBASE), pp. 282-285. IEEE, 2023

[18] pokhrel, Sangita. “LLM Based PDF Summarizer and Q/A App Using OpenAI, LangChain, and Streamlit.” Medium, February 26, 2024. https://medium.com/@sangitapokhrel911/llm-based-pdf-summarizer-and-q-a-app-using-openai-langchain-and-streamlit-807b9b133d9c.