Development and Evaluation of a Literature-Specific Chatbot

Jaideep Siva Senthil

Wayne State University, ho5863@wayne.edu

*Abstract* – Abstract - In an era where AI and ML are revolutionizing various aspects of daily life, chatbots have emerged as prominent tools in digital learning environments. This paper focuses on the development of an advanced chatbot, designed to aid users in comprehending and interacting with textbooks, acting as an effective learning assistant. Leveraging LangChain, LLM’s, and FAISS vector storage, this chatbot aims to address the specific needs of learners by providing contextual and nuanced understanding of academic texts. This approach aims to transcend the conventional limitations of chatbots, like biases and limited contextual awareness, making it particularly adept at handling complex, subject-specific queries. This approach aims to combine the newest innovations in natural language processing to create contextually aware chatbots that can assist in learning complex and niche domains with relative ease and confidence.

*Index Terms* – Chatbots, LangChain, LLMs, FAISS vector

Introduction

The evolution of chatbots marks a significant milestone in human-computer interaction, characterized by their ability to emulate human conversation through textual, auditory, and, in more advanced cases, visual means. In recent years, AI chatbot agents such as Siri, Alexa, and ChatGPT have become integral to many homes, enhancing the quality of life and redefining the role of chatbots. The advancements in artificial intelligence (AI) and machine learning (ML) have been nearly exponential, greatly increasing the sophistication and popularity of chatbots.

The history of modern chatbots can be traced back to the 1950s and 1960s, beginning with Alan Turing's groundbreaking Turing Test and the development of ELIZA by Joseph Weinbaum in 1966, a pioneering chatbot that simulated a Rogerian psychotherapist. Initially, most chatbots were rule-based, functioning as simple algorithms designed to generate predefined texts based on specific conditions. The evolution of chatbots has progressed through various stages, including the rule-based systems of the 1980s and 1990s, the widespread adoption of internet chatbots in the 2000s, and significant advancements in the 2010s. This latest phase has been marked by the integration of AI and ML technologies, significantly enhancing their natural language processing capabilities.

Despite significant technological advances, contemporary chatbots still face several challenges. Current AI/ML chatbots are often limited by their dependence on pretrained models and knowledge that is confined to their training timeline, leading to intrinsic biases. These biases, including cultural, linguistic, gender, racial, content recommendation, and ideological biases, are embedded within the models as they reflect prevailing ideologies, thus mirroring real-world flaws. Such biases, frequently a result of dominant narratives in internet-sourced training data, present substantial obstacles in achieving fair and accurate representation of diverse human [1]. Additionally, issues like verbosity, repetitive phrasing, and lack of contextual awareness in chatbots underscore the need for further improvements in generating more concise, contextually relevant, and varied responses [2]

This report introduces implementation of novel approaches to address the challenge of limited contextual awareness in niche domains, a significant issue in the development of modern chatbots. In specialized fields where texts and resources are sparse and advanced, large language models (LLMs) often struggle to procure accurate information. This report resorts to the use of LangChain to enhance chatbots' contextual understanding, the implementation of advanced embedding storage methods for more efficient data handling, and the development of domain specific LLMs for deeper domain expertise. The integration of LangChain allows for more nuanced and relevant chatbot interactions. Advanced embedding techniques improve the storage and retrieval of linguistic data. Lastly, the utilization of specialized LLMs offers a more precise understanding of specific fields. This multi-faceted approach aims to create more contextually aware chatbots that can significantly aid learning processes in fields like medicine, law, or technical sciences, where accuracy and contextual understanding are crucial.

Relevant literature

This section reviews the literature pertinent to the foundational aspect of proposed chat bot which enhances LLMs generation capability on literature specific documents using Retrieval-Augmented Generation (RAG) in language models.

I. Enhancing contextual Awareness.

The RAG framework has been instrumental in improving the contextual awareness of language models. By augmenting a pre-trained sequence-to-sequence model with a dense vector index of Wikipedia, RAG models have demonstrated a marked improvement in understanding and generating contextually relevant information [4].

II. Factuality and specificity

RAG models have outperformed baseline models in generating more factually accurate and specific responses. In generation tasks, such as defining terms and answering complex questions, RAG models have shown a notable reduction in the production of factually incorrect information, as evidenced by the data presented in Fig. 1.

A black and white text on a white background

Description automatically generated

Fig. 1. Factual information representation by RAG models

III. Application of knowledge intensive tasks

In knowledge-intensive NLP tasks, the integration of RAG has been pivotal. The study's findings suggest that for tasks such as open-domain question answering and Jeopardy question generation, RAG models have surpassed traditional seq2seq models, indicating the advantage of combining parametric and non-parametric memory systems [4]

IV. Human Assessments of model performance

Human evaluations of model performance have reinforced these findings. As shown in Fig. 2. , RAG models have been assessed to provide better factuality and specificity compared to BART, a leading seq2seq model. This is a significant testament to the effectiveness of RAG in enhancing the quality of content generated by language models.

A table with text and numbers

Description automatically generated

Fig. 2. Human assessment of models

V. Human Assessments of model performance

Furthermore, RAG models have been evaluated against task-specific architectures, with the research highlighting their superior performance. This is particularly relevant in fields where accurate and precise information retrieval is crucial, showcasing the versatility of RAG models.

Methodology

I. LangChain

In the pursuit of creating language models that exhibit a profound understanding of domain-specific literature, the implementation of LangChain provides a significant leap forward. LangChain, a library designed to facilitate the chaining of language model outputs with external computation and data retrieval, offers a novel mechanism to enhance the capabilities of language models, especially in processing and generating domain-specific content. By enabling language models to query databases, invoke APIs, and perform logical reasoning, LangChain acts as an intermediary that extends the base functionality of models like RAG. This allows for the creation of systems that not only generate more contextually aware responses but can also interact with and utilize domain-specific resources in real-time. For instance, in a legal advisory chatbot, LangChain could empower the model to pull the latest case law or statutes relevant to a user's query, thereby providing responses that are not just accurate in terms of legal terminology but are also current and practical. Leveraging LangChain in model implementation ensures that the system's output is not only informed by a vast repository of information but is also dynamically aligned with the evolving landscape of the domain-specific literature.

II. FAISS structure

The efficient handling of vector data is critical for the performance of Retrieval-Augmented Generation (RAG) models, particularly in retrieving relevant information from vast datasets. The FAISS (Facebook AI Similarity Search) vector storage system emerges as a highly efficient solution for this task. Developed by Facebook AI Research, FAISS is optimized for the rapid search of similarity in large-scale vector databases, making it an invaluable component in the RAG architecture. It allows for the quick retrieval of information by conducting nearest neighbor searches in the embedding space, which is essential for the RAG's non-parametric memory component. This search capability enables RAG models to access the most relevant pieces of information almost instantaneously, significantly enhancing the model's ability to generate accurate and contextually appropriate responses. FAISS's scalability and speed are particularly beneficial when working with domain-specific literature, where the ability to swiftly sift through specialized information can determine the precision and usefulness of the model's output. By integrating FAISS within RAG models, developers can leverage its robust indexing and retrieval features to build powerful language models capable of navigating complex informational landscapes with unprecedented efficiency.

III. Bloom LLM

BLOOM, a large language model developed by a collaborative effort known as Big Science, has been specifically architected to cater to the nuanced requirements of STEM (Science, Technology, Engineering, and Mathematics) and niche domains. Its design is predicated on a diverse and expansive training dataset that includes scientific papers, technical documentation, and specialized domain literature, which imbues BLOOM with an intrinsic understanding of complex, subject-specific vocabulary, and concepts. This extensive training enables BLOOM to perform exceptionally well in generating and interpreting technical content, making it an ideal choice for STEM-related applications. Moreover, BLOOM's capacity to comprehend and process data from less-represented languages and specialized jargon addresses the common challenge of language models lacking in areas with limited data availability. Its multilingual capabilities ensure that knowledge and expertise can be accessed and leveraged across linguistic barriers, an essential feature for global collaboration in STEM fields. Additionally, BLOOM's open-source nature allows for continual refinement and customization, empowering researchers and practitioners to fine-tune the model for highly specialized tasks. This adaptability is crucial for niche domains where off-the-shelf models often fall short. In summary, BLOOM's comprehensive training, multilingual support, and open-source accessibility render it a powerful tool for advancing research and development in STEM and other specialized fields, where precision, depth of knowledge, and technical accuracy are paramount.

IV. Overall architecture for chatbot

A diagram of a chatbot model

Description automatically generated

Fig. 3. Chatbot architecture

Incorporating the integration of advanced AI technologies, our domain-specific chatbot model capitalizes on the strengths of BLOOM LLM, FAISS vector storage, and LangChain to deliver an exceptional user experience tailored to specialized fields. The model commences by deconstructing comprehensive domain documents into manageable segments, a process that benefits from BLOOM LLM's adeptness at parsing and understanding intricate subject matter. These segments are subsequently converted into embeddings, with FAISS providing an optimized vector storage and retrieval framework, ensuring rapid and accurate matching of these embeddings against user queries. LangChain enhances the chatbot's capability to utilize these embeddings to extract pertinent information from the knowledge database, enabling the generation of responses that are not only contextually aware but also dynamically informed by the latest domain-specific insights. Through this iterative cycle of query handling and information retrieval, the chatbot continuously refines its knowledge base, ensuring that each interaction is informed by up-to-date and relevant content. This model represents a convergence of state-of-the-art AI components, each contributing to a system that is designed to evolve, learn, and provide increasingly sophisticated domain-specific guidance and support.

Methodology

The experimental setup for the domain-specific chatbot model involved a comprehensive implementation pipeline utilizing state-of-the-art technologies and libraries. The implementation was facilitated by a Python environment, where several packages were installed, including LangChain, TikToken, PyPDF, FAISS, Cohere, OpenAI, Transformers, Accelerate, Bitsandbytes, Xformers, and Einops for optimization and execution.

I. Installation and Dependencies

The following Python libraries were installed to support the model's functionality:

* LangChain: For chaining language model outputs with external computation and data retrieval.
* FAISS: A library for efficient similarity search and clustering of dense vectors.
* Transformers and InstructorEmbedding: For text embeddings and model integration.
* Sentence Transformers: For creating sentence-level embeddings.

II. Model and Library Importation

Python warnings were suppressed for cleaner output, and essential libraries were imported, including those for document loading, text splitting, prompting, vector storage, and model retrieval. The HuggingFacePipeline and HuggingFaceInstructEmbeddings from the Transformers library were used alongside LangChain’s RetrievalQA for question-answering tasks.

III. Configuration and Model Selection

A configuration class, Config, was defined to set parameters for the language models, text splitting, and embeddings. The BLOOM model, specifically 'bigscience/bloom-7b1', was selected for its ability to generate and interpret domain-specific content. It was configured with specific parameters including temperature control, top\_p for controlling the randomness of predictions, and a repetition penalty to avoid redundant information.

IV. Model Initialization and Configuration

The BLOOM model was instantiated with optimization for GPU usage, utilizing 4-bit inference and automatic device mapping to balance the load between CPU and GPU, ensuring efficient resource utilization. The model was configured to perform with low CPU memory usage and a maximum length of 1024 tokens.

V. Document Handling and Embeddings

Documents were sourced from a specified directory and split into smaller chunks using a RecursiveCharacterTextSplitter with predefined chunk size and overlap settings. Embeddings for these chunks were created using a sentence-transformers model repository, enabling efficient information retrieval and matching for incoming user queries.

VI. Vector Storage and Retrieval

A FAISS vector store was established to facilitate the rapid retrieval of similar passages from the knowledge database, leveraging nearest neighbor searches to enhance the chatbot's response accuracy and relevancy.

VII. Question and Answer Chain

The RetrievalQA chain from LangChain was integrated to enable the chatbot to handle complex question-answering tasks, utilizing the combination of the BLOOM model's comprehension and the efficiency of the FAISS vector store.

Results

Results were analysis of the text generated in relation to how true the information being provided is in comparison to how specific it is to the question asked. Four questions were asked before FAISS vector similarity was given to the BLOOM model:

I. Questions before contextual awareness

* What is relation extraction?
* Please explain relation extraction in simple words
* What is perplexity?
* Please explain perplexity in simple words

Six questions were asked after the FAISS vector database similarity was provided to the LLM model to generate information.

I. Questions after contextual awareness

* What is relation extraction?
* Please explain relation extraction in simple words
* What is perplexity?
* Please explain perplexity in simple words
* Give me R code for understanding decision tree
* What is the best dinner place in Detroit

The last two questions were asked to determine how the model handled OOV questions. The chat bot was then scored on factuality and specificity, with 1 being mostly accurate, 0.5 being partially accurate and 0 being irrelevant. The results from before and after the contextual awareness is presented in Table. 1.

|  |  |  |
| --- | --- | --- |
|  | Factuality | Specificity |
| Before contextual awareness | | |
| Q1 | 1 | 1 |
| Q2 | 0 | 0 |
| Q3 | 0 | 0 |
| Q4 | 0 | .5 |
| After contextual awareness | | |
| Q1 | 1 | 1 |
| Q2 | 1 | 1 |
| Q3 | 1 | 1 |
| Q4 | .5 | 1 |
| Q5 | 0 | 0 |
| Q6 | 0 | 0 |

Table. 1. Factuality and specificity score for chat bot

Discussion

The results of our experiment offer valuable insights into the efficacy of integrating cutting-edge AI technologies in the development of domain-specific chatbots. The improved text generation rate in responses underscores the BLOOM LLM's capacity for a deep understanding of specialized lexicons with the context of specific literature, while the swift response times achieved are indicative of the efficiency afforded by the FAISS vector storage system. These outcomes affirm the idea that sophisticated retrieval mechanisms, when combined with contextually aware language models, can elevate the performance of chatbots within specialized domains.

The efficacy of the model in niche areas is particularly noteworthy, with LangChain enabling real-time dynamic querying and retrieval that is highly pertinent to user inquiries. This responsiveness to context is a testament to the potential of RAG to transform service levels in sectors where specialized knowledge is at a premium. algorithms to further enhance the user experience.

When juxtaposed with existing technologies, the improvements observed signal a significant advancement. The chatbot's proficiency suggests that leveraging domain-specific embeddings and real-time retrieval is a successful strategy to address some of the conventional limitations of chatbot technologies. However, the chat bot was not without its challenges some of which being the answer process can take a long time with the average time for query response being around 30s, the chat bot does not handle out of context enquires very well, providing often incorrect and irrelevant answers.

Conclusion

Looking ahead, the aim is to broaden the knowledge base, integrate a more varied set of data sources, and improve the retrieval algorithms to manage an expanded array of queries more effectively. Investigating the incorporation of feedback mechanisms could offer a pathway for the chatbot to learn iteratively from user interactions, thereby continuously refining its knowledge and performance.

In conclusion, the findings from this study are promising and represent a feasible approach in the realm of domain-specific chatbots. With the rapid progress in artificial intelligence, the applications for such models are vast, opening possibilities for revolutionizing interactions with information systems across various sectors and providing individualized AI assistants that can improve the quality of life.

References

[1] Buolamwini, J., & Gebru, T. (2018). Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. Proceedings of Machine Learning Research, 81, 1–15.

[2] Chowdhury, G. G. (2020). Natural Language Processing. Annual Review of Information Science and Technology, 37(1), 51–89.

[3] Milgram, P., & Kishino, F. (1994). A Taxonomy of Mixed Reality Visual Displays. IEICE Transactions on Information and Systems, E77-D(12).

[4] P. Lewis, “Retrieval-Augmented Generation for Knowledge-Intensive NLP tasks,” *arXiv.org*, May 22, 2020. <https://arxiv.org/abs/2005.11401v4>

[5] “Empowering domain adapted LLM’s.” <https://www.arcee.ai/blog/the-pros-and-cons-of-retrieval-augmented-generation-and-fine-tuning-with-llms>

[6] K. Martineau, “What is retrieval-augmented generation?,” *IBM Research Blog*, Oct. 20, 2023. <https://research.ibm.com/blog/retrieval-augmented-generation-RAG>

[7] Bigscience-Workshop, “GitHub - bigscience-workshop/bigscience: Central place for the engineering/scaling WG: documentation, SLURM scripts and logs, compute environment and data.,” *GitHub*. <https://github.com/bigscience-workshop/bigscience/tree/master>

[8] “Introduction| ️🔗Langchain.” https://python.langchain.com/docs/get\_started/introduction