Context-Driven Chatbot Development: Leveraging Zephyr-7b with RAG for Improved Response Accuracy

Sandra Jervas
Centre for AI
TKM College of Engineering
Kollam, India
sandrajervas206@gmail.com

Chinnu Jacob
Centre for AI
TKM College of Engineering
Kollam, India
chinnujacob@tkmce.ac.in

Sumod Sundar
Centre for AI
TKM College of Engineering
Kollam, India
sumodsundar@tkmce.ac.in

Deepasree Varma P
Knowledge Office
ICT Academy of Kerala
Trivandrum, India
knowledge.officer5@ictkerala.org

2025 Emerging Technologies for Intelligent Systems (ETIS) | 979-8-3315-0754-1/25/$31.00 ©2025 IEEE | DOI: 10.1109/ETIS64005.2025.10960935

***Abstract*—In recent years, the demand for AI-driven conversational agents has increased significantly across various industries. Traditional generative models, while capable of producing human-like responses, often struggle with factual accuracy and context retention. Retrieval Augmented Generation (RAG) presents a novel approach to enhance the performance of AI chatbots by combining the strengths of both retrieval-based and generative models. The paper focuses on the development of an AI-driven chatbot using a RAG framework, integrating the Zephyr-7b model, to serve the needs of the ICT Academy of Kerala (ICTAK). The chatbot is engineered to deliver precise and contextually relevant responses to user queries by integrating advanced language models with structured data extracted from the ICTAK website. The data, which was meticulously scraped and processed, ensures that the chatbot's knowledge base remains current and accurately reflects the ICTAK's services and operations. A critical challenge addressed by this work is the issue of hallucinations in large language models (LLMs), where models may generate seemingly plausible yet incorrect or irrelevant information. To counteract this, the paper employs various chunking methods that enhance the chatbot's capability to retrieve and generate accurate responses. This AI Chatbot incorporates cutting-edge techniques in natural language processing, including document retrieval, context compression, and re-ranking. This multi-faceted approach ensures that the chatbot not only provides accurate responses but also delivers them in a contextually relevant manner.**

***Index Terms*—LLM, chatbot, hallucinations, RAG, Zephyr-7b**

I. Introduction

In the rapidly evolving field of Natural Language Processing (NLP), the development of advanced conversational agents, particularly chatbots, has garnered significant attention. These systems are designed to simulate human-like interactions, offering users timely and relevant responses based on their queries. The integration of NLP in chatbots aims to enhance user experience, provide real-time assistance, and facilitate access to information in a conversational manner [1], [2]. As digital transformation continues to shape various industries, the demand for efficient and intelligent chatbots has surged. These systems not only streamline communication but also enable organizations to manage large volumes of queries effectively. For educational institutions, such as the ICTAK, deploying a chatbot can significantly improve user engagement by providing instant responses to frequently asked questions and offering personalized assistance [3], [4]. In the context of educational settings, chatbots can address the limitations of traditional information dissemination methods. Manual handling of queries is often labor-intensive and prone to delays. By leveraging automated systems, educational institutions can enhance their operational efficiency, ensure timely communication, and deliver consistent responses. This shift from manual to automated interactions is crucial in managing the increasing demand for support and information in educational environments [5], [6]. The development of a robust chatbot involves several key components, including effective data retrieval, accurate response generation, and the handling of various user queries. The proposed work focuses on creating an AI-based chatbot using RAG framework. This approach combines the strengths of information retrieval and generative models to provide contextually accurate and coherent responses [7], [8].

The chatbot's foundation relies on data scraped from the ICTAK website, which has been meticulously structured and processed. This dataset serves as the basis for the chatbot's knowledge base, enabling it to retrieve relevant information and generate responses. The challenge of hallucinations in LLMs is addressed through advanced chunking methods, which ensure that the generated responses are grounded in the provided context and reduce the likelihood of generating irrelevant or incorrect answers. By employing state-of-the-art techniques in NLP and leveraging the structured data from ICTAK, this work aims to develop a highly effective chatbot that can provide accurate, contextually relevant, and user-friendly interactions. The goal is to create a system that not only enhances the user experience but also supports the institution's efforts in managing communication and information dissemination efficiently. Additionally, these chatbots can be easily scaled to handle diverse and complex queries, making them an indispensable tool for institutions

looking to provide seamless and personalized interactions with minimal manual intervention.

The key contributions of the paper are as follows:

- Development of a chatbot using the RAG framework with Zephyr-7b model for enhanced user interaction and accuracy.
- Creation of a robust knowledge base from data scraped from the ICTAK website, ensuring up-to-date information.
- Implementation of advanced NLP techniques such as document retrieval, context compression, and re-ranking for accurate, contextually relevant responses.
- Demonstration of how the chatbot improves user engagement by providing instant responses and personalized assistance for educational institutions.

Rest of this paper is organized as follows. Section II presents the literature survey. Section III describes the proposed methodology that includes data collection and preprocessing, model development and also evaluation and performance metrics. Section IV presents the implementation in detail. Section V presents the results and discussions, and Section VI concludes.

II. Related Works

Recent advancements in AI have profoundly impacted the processing and utilization of healthcare data. Notably, the application of advanced AI techniques, including generative models and RAG, has gained traction for enhancing clinical decision support and information retrieval. Generative AI models have shown considerable promise in transforming the handling of Electronic Health Records (EHRs). Alkhalaf et al. [1] explored the application of zero-shot prompt engineering in combination with RAG to automate the summarization and extraction of malnutrition-related information from EHRs in residential aged care facilities. The authors utilized the Llama 2 13B model [9] and developed a malnutrition-specific labeled dataset. Despite challenges related to data format inconsistencies and medical terminology, the research demonstrated the potential of generative AI in creating structured summaries from unstructured clinical notes. Improving document retrieval from Electronic Medical Records (EMRs) has been advanced by learning-to-rank approaches. In another study, a learning-to-rank system is investigated and designed to enhance RAG-based search engines by incorporating semantic context and user feedback [2]. The system employed semantic embeddings and LLMs such as GPT 3.5 [10] to refine search algorithms. This approach improved the accuracy of document retrieval and highlighted the benefits of combining advanced ranking techniques with user-centric feedback for better clinical decision support. The concept of RAG has been applied to knowledge-intensive NLP tasks with notable success. Lewis et al. [3] examined the RAG model's integration of a pre-trained sequence-to-sequence model (BART) [11] with a dense vector index accessed via a Dense Passage Retriever (DPR). This hybrid approach enables the generation of more informed responses by conditioning outputs on relevant external documents. The findings demonstrated RAG's superiority over traditional parametric-only models, particularly in open-domain question answering. A comprehensive survey on RAG methodologies conducted by Gao et al. [4] categorized the evolution of RAG into Naive, Advanced, and Modular paradigms. Naive RAG involves a basic process of indexing, retrieval, and generation, while Advanced RAG introduces improvements in retrieval quality through enhanced pre-retrieval and post-retrieval processes. Modular RAG further advances the field by integrating specialized modules and interaction patterns, allowing for greater flexibility and effectiveness. This survey underscores the growing significance of RAG in various applications, including healthcare. Accurate evaluation of retrieval models within RAG systems remains a challenge. The innovative eRAG approach developed by Salemi and Zamani [5] addresses this by evaluating each document's contribution to downstream tasks, as processed by the LLM within the RAG system. Traditional retrieval evaluation methods showed limited correlation with downstream performance, but eRAG provided a more accurate reflection by comparing LLM-generated outputs with ground truth labels. This method demonstrated significant improvements in correlating retrieval performance with downstream metrics. Thulke et al. [6] focused on improving the efficiency of RAG in task-oriented dialogue systems. Various methods, including hierarchical selection and Dense Knowledge Retrieval with a siamese network structure, were proposed and tested on the MultiWOZ 2.1 dataset. These approaches aimed to streamline turn detection, knowledge selection, and response generation, demonstrating the effectiveness of retrieval-augmented methods in enhancing the quality and efficiency of task-oriented dialogue systems. Shi et al. [7] enhanced RAG models by incorporating user profiles for improved personalization and efficiency. The authors compared a basic LLM Reader with a Personalized LLM Reader, assessing differences in response quality and overall effectiveness. Using pairwise comparisons and dynamic user profiles, the research highlighted the benefits of personalization in optimizing response generation. Additionally, the study analyzed efficiency by adjusting similarity thresholds in the Retrieval Trigger module to balance response time and quality. Further analysis of the ERAGent framework focused on response generation and personalization. The comparative analysis between versions with and without user profiles utilized LLMs as judges for evaluating response quality. The authors emphasized the importance of personalization and efficient interaction, highlighting how user profiles and adjustable similarity thresholds contribute to improved response generation and overall system performance. Dense Passage Retrieval (DPR) has been a significant advancement for open-domain question answering. The methodology described by Karpukhin et al. [8] involves a dual-encoder architecture for questions and passages, trained using in-batch negative sampling. The study evaluated DPR's performance against various benchmarks, demonstrating its effectiveness in

improving retrieval accuracy compared to traditional methods. The focus on sample efficiency and robustness underscores DPR's value in handling diverse question-answering tasks. The training and efficiency of DPR were explored by Karpukhin et al. [8], detailing the model's dual-encoder architecture and training scheme. The authors examined sample efficiency and performance across various benchmarks, illustrating DPR's effectiveness in open-domain question answering. This research highlights DPR's robustness and efficiency, making it a valuable tool for handling complex question-answering tasks.

III. METHODOLOGY

For the RAG system, the goal is to generate accurate and contextually relevant responses to user queries by leveraging a combination of retrieval and generation techniques. Unlike traditional response systems that rely solely on predefined answers, this RAG system dynamically retrieves relevant information from a knowledge base and generates responses based on that context. By integrating advanced models and methodologies, including web scraping, contextual compression, and LLMs, the system ensures that responses are both precise and informative. Here's a detailed description of the RAG framework as shown in Figure 1:

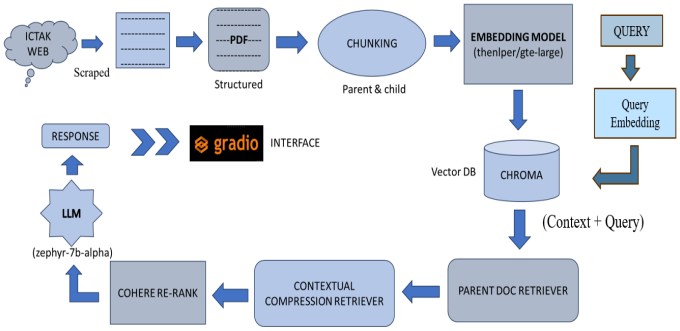

Fig. 1. Proposed framework for RAG-based system

A. Data Collection and Preprocessing

1) Web Scraping: Data is collected from the ICTAK website to create a dataset for further processing. The raw data is scraped from ICTAK web pages, collecting text and structuring the information for use in the chatbot.
2) Data Preprocessing: The scraped data undergoes preprocessing to remove any irrelevant content and structure it into a standardized format (e.g., converting into PDF). This step ensures that the data is suitable for chunking and embedding, providing a clean and organized dataset for the next phases.
3) Chunking (Parent and Child): The structured data is divided into "Parent" and "Child" chunks, as shown in Figure 2. Parent chunks represent larger sections, while Child chunks are smaller, more focused portions within each Parent chunk. This approach facilitates efficient retrieval of relevant information later in the process.

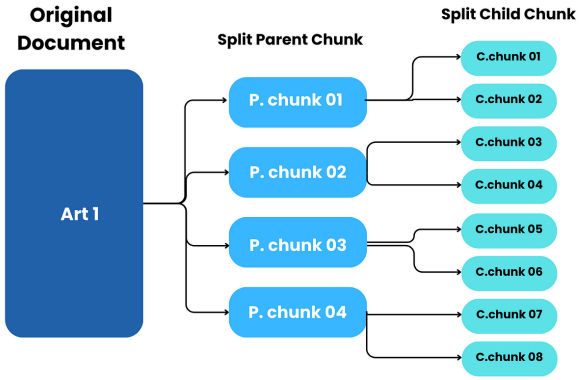

Fig. 2. Chunking Process [12]

B. Model Development

1) Embedding Model (Hugging Face - thenlper/gte-large): The text chunks (both Parent and Child) are passed through the embedding model, transforming them into vector embeddings. These embeddings capture the semantic meaning of the text, which is essential for effective retrieval and ranking.
2) Vector Store (Chroma): The vector embeddings are stored in a Vector Database (Chroma). This setup allows for efficient searching and retrieval based on similarity to a user query, playing a critical role in finding the relevant context for generating responses.
3) Parent Document Retriever: When a user query is initiated, the Parent Document Retriever searches the vector database to find relevant Parent chunks containing useful information related to the query.
4) Contextual Compression Retriever: The retrieved documents are compressed to highlight the most relevant information. This ensures that only the most pertinent data is considered for final response generation, refining the context provided to the model.
5) Cohere Re-Rank: After compression, the documents are reranked using the Cohere model, which prioritizes the most relevant context. As shown in Figure 3, this step guarantees that the model focuses on the most useful information for generating accurate responses.

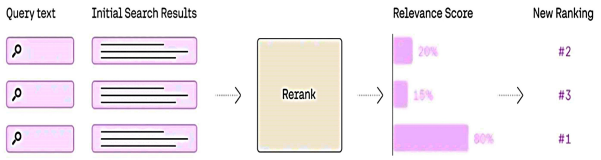

Fig. 3. Re-Ranking Process [13]

6) LLM (Language Model - Zephyr-7b-alpha): The LLM takes the user query along with the most relevant,

reranked context to generate a coherent and accurate response. The model leverages its understanding of the context and the user's input to produce meaningful outputs.

7) Gradio Interface: The final response generated by the LLM is displayed to the user through the Gradio interface. This user-friendly environment allows users to input their queries and view responses seamlessly.

C. Evaluation and Performance Metrics

1) Testing Procedure: The RAG chatbot is tested with various queries to evaluate its performance. Different scenarios are created to assess how well the system handles a range of inquiries, from basic questions to more complex queries. This ensures comprehensive evaluation across diverse contexts and domains.
2) ROUGE Score Analysis: To further evaluate the performance of the chatbot, ROUGE (Recall-Oriented Understudy for Gisting Evaluation) scores are calculated for the generated responses. These scores compare the similarity between the chatbot's output and the reference text in terms of precision, recall, and F1-score. The ROUGE score provides insights into how well the chatbot captures the key phrases and ideas from the reference responses, helping to fine-tune the model for better alignment and relevance.

IV. Implementation

A. Experimental Setup

To efficiently perform the programming tasks and execute the AI models, Google Colab, a cloud-based platform, was utilized. Google Colab provides an accessible environment with GPU resources, enabling accelerated computation and seamless Python programming for large-scale language model training and testing. For the implementation of the Retrieval-Augmented Generation (RAG) framework, the following key libraries and frameworks were employed:

- Hugging Face: The pre-trained language models and tokenizers were sourced from the Hugging Face model hub. Specifically, the Zephyr-7b-alpha model was integrated using the Hugging Face Transformers library, generation of responses in context with user queries.
- Langchain: The Langchain framework was used to integrate the various components of the RAG system, including document retrieval, chunking, and the overall orchestration of the query-to-response pipeline. Langchain's flexibility allowed for seamless integration with other tools like vector databases and embedding models.
- Chroma: To manage the vector embeddings of the chunked documents, Chroma was used as the vector database. It facilitated the efficient retrieval of relevant chunks based on their semantic similarity to the user's query, contributing to the improved accuracy of the generated responses.
- Cohere: For re-ranking the compressed documents, the Cohere model was employed. This model ensured that the most contextually relevant information was prioritized before being passed to the language model for final response generation, enhancing the system's overall performance.
- Gradio: The final user interface was built using Gradio, which provided a user-friendly platform where users could input their queries and receive responses in real-time. Gradio's simplicity and integration capabilities allowed for rapid deployment of the chatbot interface.

By utilizing these advanced frameworks and tools, the system was able to efficiently handle the complexities of retrieval, ranking, and generation while maintaining high accuracy and relevance in the responses.

B. Data Preparation

The experiment utilized a dataset initially web scraped, then structured, and saved as a PDF file. This dataset also included a detailed syllabus provided by ICTAK, which was integrated into the formatted scraped data. The data was loaded using the UnstructuredPDFLoader from LangChain. The document was then split into manageable chunks using two levels of text splitting: Parent Splitter (Split text into chunks of 512 characters and Child Splitter (Further split these chunks into smaller segments of 100 characters. This hierarchical splitting approach was designed to balance the context size and the granularity of the retrieved documents, ensuring that both the detailed syllabus and other content from the dataset were effectively captured for the RAG model.

C. Embeddings and Vector Store Setup

Embeddings for the document chunks were created using the Hugging Face Inference API Embeddings with the model "thenlper/gte-large" [14]. These embeddings were stored in a Chroma vector store, which facilitates efficient retrieval of relevant chunks based on the query.

D. Retriever and Compression

The Parent Document Retriever was set up with the following components: Vectorstore (For storing and querying document embeddings) and an InMemoryStore (For holding document metadata). The retriever was responsible for fetching relevant chunks from the vector store based on a given query. A Contextual Compression Retriever [15] was employed to refine the retrieved results. This was achieved using a custom reranker Custom Cohere Rerank [16], leveraging the Cohere API for contextual compression to improve the relevance of the retrieved context.

E. Model and Prompt Setup

The RAG model used for generating responses was the HuggingFaceHub model "HuggingFaceH4/zephyr-7b-alpha" [17]. A ChatPrompt Template was created to format the context and query for the model, ensuring that responses were relevant and accurately reflected the content of the dataset.

F. Interface and Deployment

The final setup included a Gradio interface for user interaction:

- Inputs: A textbox for entering questions.
- Outputs: A textbox for displaying responses.
- Title: "ICTAK Assistant" The interface provided a user-friendly way to query the chatbot and receive responses based on the context retrieved from the dataset.

V. Results and Discussions

The RAG chatbot was tested with various queries to evaluate its performance. The effectiveness of the system was assessed based on the accuracy and relevance of the responses. One of the test queries, as shown in Figure 4, asked the chatbot to "Give details about the Python programming course." In response, the chatbot accurately provided information, stating that it is a 2-month course covering Python basics, database concepts, and web development. The answer was coherent, contextually relevant, and met the user's expectations for detailed information about the course.

In another test case, presented in Figure 5, the query was: "I am an MTech in AI student, what should I do?" The chatbot's response was insightful, recommending that the user explore advanced AI-related courses. Specifically, it suggested the Postgraduate Program (PGP) in Data Science and Machine Learning, highlighting its relevance for students pursuing AI.

This personalized recommendation demonstrated the chatbot's ability to adapt its response to the user's academic background, providing value-added suggestions for further education. The use of data compression and document reranking significantly contributed to the improvement in response quality, allowing the chatbot to prioritize and focus on the most relevant pieces of information. These techniques ensured that the chatbot's generated responses were not only accurate but also contextually aligned with the user's queries.

As evidenced by the responses in Figures 4 and 5, the system's performance was evaluated on how well it could understand and generate responses tailored to user inputs. The chatbot consistently delivered precise and helpful answers, illustrating the efficacy of the Retrieval-Augmented Generation (RAG) framework in enhancing chatbot interaction. These results validate the effectiveness of the RAG approach in handling diverse queries and providing useful responses based on the dataset.

To quantitatively assess the chatbot's performance, ROUGE scores were calculated for five test queries. These scores provide a measure of how well the generated responses align with reference answers in terms of n-grams (unigrams, bigrams) and longest common subsequences.

The following test queries were used to calculate the ROUGE scores:

- Query 1: "Give details about the Python programming course."
- Query 2: "I am an MTech in AI student, what should I do?"
- Query 3: "What are the courses available?"
- Query 4: "How to apply for the courses provided by ICTAK?"
- Query 5: "What topics are covered in a course on Machine Learning and Artificial Intelligence?"

The table I below presents the ROUGE scores for these five test queries:

TABLE I
ROUGE Scores for Test Queries

Query ID	ROUGE-1	ROUGE-2	ROUGE-L
Query 1	0.7012	0.6942	0.6874
Query 2	0.8321	0.8012	0.8256
Query 3	0.7894	0.7577	0.7650
Query 4	0.6617	0.6321	0.6578
Query 5	0.7681	0.6926	0.7394

These scores indicate that the chatbot's responses were generally well-aligned with the reference answers, with strong recall and precision in most cases. Specifically:

- **ROUGE-1** measures the overlap of unigrams (individual words) between the generated response and the reference.
- **ROUGE-2** measures the overlap of bigrams (pairs of words). .
- **ROUGE-L** evaluates the longest common subsequence between the generated response and the reference.

The table highlights that, overall, the chatbot's responses exhibited good performance in terms of matching the reference answers. The use of document reranking and data compression techniques, as mentioned earlier, helped the chatbot focus on the most relevant content, which likely contributed to these high ROUGE scores.

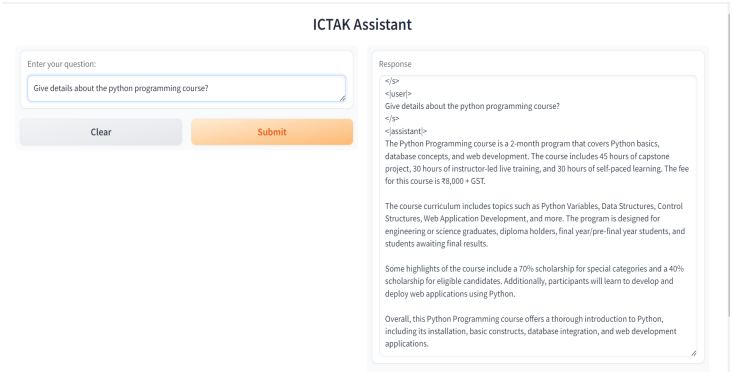

Fig. 4. Response (Deployed using Gradio)

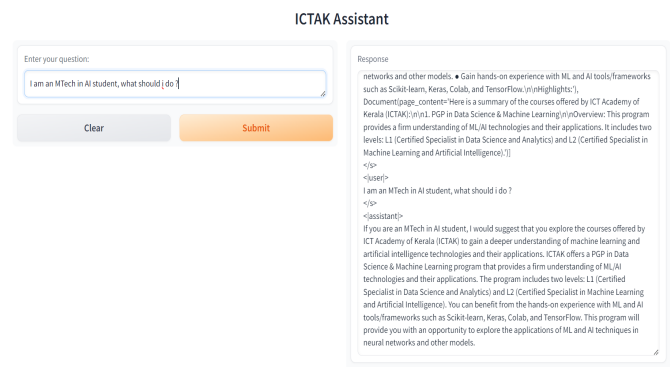

Fig. 5. Response Example 2

VI. Conclusion

In the analysis of the RAG-based system, it was observed that the integration of retrieval and generation techniques yielded highly accurate and contextually relevant responses, making it one of the most effective methods for dynamic information retrieval and response generation. The use of Chroma for vector storage, combined with the Cohere re-ranking mechanism, allowed the system to prioritize the most relevant documents, enhancing the quality of the final output generated by the language model. However, challenges such as the potential for overfitting in the language model and the dependency on the quality of the initial web-scraped data were identified, highlighting areas for further improvement.

Looking ahead, there is a significant opportunity to explore transformer-based architectures, such as BERT or GPT, to enhance the system's capabilities further. These models have shown exceptional proficiency in natural language understanding and generation tasks, and their application within the RAG framework could lead to even more accurate and contextually appropriate responses. Additionally, expanding the system to handle domain-specific datasets, such as those in the medical or legal fields, could significantly broaden its applicability and usefulness.

Furthermore, the development of a more sophisticated contextual compression mechanism, perhaps leveraging advanced transformer models, could improve the system's ability to con- dense large volumes of data into the most relevant information. This would not only enhance response accuracy but also reduce computational overhead, making the system more efficient. The future scope also includes the possibility of creating domain-specific RAG systems that are fine-tuned for particular industries or fields, thereby providing highly specialized and accurate information retrieval and generation services.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to the Information and Communication Technology Academy of Kerala (ICTAK) for their support and encouragement throughout this research. Their commitment to promoting technology and innovation has significantly contributed to the development of this work.

References

[1] Mohammad Alkhalaf, Ping Yu, Mengyang Yin, and Chao Deng. Applying generative ai with retrieval augmented generation to summarize and extract key clinical information from electronic health records. *Journal of Biomedical Informatics*, page 104662, 2024.

[2] Cheng Ye. Exploring a learning-to-rank approach to enhance the retrieval augmented generation (rag)-based electronic medical records search engines. *Informatics and Health*, 1(2):93–99, 2024.

[3] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[4] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[5] Alireza Salemi and Hamed Zamani. Evaluating retrieval quality in retrieval-augmented generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2395–2400, 2024.

[6] David Thulke, Nico Daheim, Christian Dugast, and Hermann Ney. Efficient retrieval augmented generation from unstructured knowledge for task-oriented dialog. *arXiv preprint arXiv:2102.04643*, 2021.

[7] Yunxiao Shi, Xing Zi, Zijing Shi, Haimin Zhang, Qiang Wu, and Min Xu. Eragent: Enhancing retrieval-augmented language models with improved accuracy, efficiency, and personalization. *arXiv preprint arXiv:2405.06683*, 2024.

[8] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020.

[9] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[10] Junjie Ye, Xuanting Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, Zeyang Zhou, Chao Gong, Yang Shen, et al. A comprehensive capability analysis of gpt-3 and gpt-3.5 series models. *arXiv preprint arXiv:2303.10420*, 2023.

[11] M Lewis. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[12] AI Insights Cobet. Rag and parent document retrievers: Making sense of complex contexts with code. https://medium.com/ai-insights-cobet/rag-and-parent-document-retrievers-making-sense-of-complex-contexts-with-code-5bd5c3474a8a, 2024. Accessed: 2024-09-26.

[13] Yasir PK. Revolutionizing search with cohere's reranking approach: A developer's guide. https://medium.com/@developer.yasir.pk/revolutionizing-search-with-coheres-reranking-approach-a-developer-s-guide-4c207d39ce94, 2024. Accessed: 2024-09-26.

[14] The NLP ER. Gte large. https://huggingface.co/thenlper/gte-large, 2024. Accessed: 2024-09-26.

[15] Sourav Verma. Contextual compression in retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2409.13385*, 2024.

[16] Cohere. Rerank. https://cohere.com/rerank, 2024. Accessed: 2024-09-26.

[17] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*, 2023.