An Intelligent Assistant for Multiformat Data Search and Retrieval Using GenAI

Dr. T. R. Lekhaaa
Associate Professor,
Department of Information Technology
SNS COLLEGE OF TECHNOLOGY
(Anna University)
Coimbatore, India
lekhaa.tr.it@snsce.ac.in

Dr. P. Sumathi
Head of the Department,
Department of Information Technology
SNS COLLEGE OF TECHNOLOGY
(Anna University)
Coimbatore, ,India
psumathi.it@gmail.com

Suresh S
Student,
Department of Information Technology
SNS COLLEGE OF TECHNOLOGY
(Anna University)
Coimbatore, India
suresh5122003@gmail.com

Priyan G
Student,
Department of Information Technology
SNS COLLEGE OF TECHNOLOGY
(Anna University)
Coimbatore, ,India
priyang2339@gmail.com

Gowtham S
Student,
Department of Information Technology
SNS COLLEGE OF TECHNOLOGY
(Anna University)
Coimbatore, India
gowthamcr786@gmail.com

Ridha Marva B
Student,
Department of Information Technology
SNS COLLEGE OF TECHNOLOGY
(Anna University)
Coimbatore, India
ridhamarva@gmail.com

***Abstract*—** With organizations now placing trust in unstructured, multimodal and multilingual data, the need for intelligent assistants that can traverse complex enterprise information landscapes has escalated at a rate that has seldom been seen before. This paper introduces a domain-agnostic enterprise assistant system built on a Retrieval-Augmented Generation (RAG) framework that leverages the latest in large language models (LLMs), all linked to a hybrid retrieval pipeline enabling complex responses to user queries and input from text, image and audio sources. While conventional AI assistants are limited to keyword-based search or unimodal interaction, our assistant encodes inputs in context and in real-time through either a sophisticated language model or a model with improved data priorization through semantic enhancement. The enterprise assistant system we propose utilizes a modular microservices architecture to enhance scalability, fault tolerance, and linking with heterogeneous enterprise infrastructure. The multimodal ingestion pipeline, chunk-level text embedding using transformers, dynamic metadata generation, and vector retrieval using state-of-the-art databases like Qdrant are just a few of the subsystems that comprise the proposed system, which incorporates user-language independence from the original data source, such as local/semantic documents, and consistent performance when performance across different languages is desired. Control of use permissions to system features is easily managed via a robust role-based access control (RBAC) approach, with tiered access capabilities for superadmins, admins, editors, and end-users. Building on recent developments in RAG pipelines, long-context modeling, and multimodal optimization frameworks, the assistant also included supportive features for contextual memory and retrieval caching to enhance latency and response association. Benchmark tests compared the assistant to baseline LLM systems and static retrieval systems, revealing impressive benefits to response accuracy, retrieval latency and multimodal consistency. To prioritize core adaptable architecture, dependencies on domains were purposely decoupled. This system is a flexible foundation for supporting intelligent automation within a wide range of enterprise environments such as the legal services industry, the manufacturing sector, education and the public good.

Keywords— Retrieval-Augmented Generation, Multimodal AI, Enterprise Assistant, Microservices Architecture, Vector Embeddings, Role-Based Access Control, Multilingual Interaction, Transformer Models.

I. INTRODUCTION

Modern enterprises have to deal with large amounts of both silky structured, and unstructured structured data, from databases to documents, images, to voice recordings, whenever they seek to leverage knowledge for decision-making. Traditional enterprise content systems are very proficient at storing content and transactional methods for retrieving that content but are not capable of understanding semantics and providing real-time intelligence for heterogeneous data processing. Today's large language models (LLMs), (e.g., GPT, LLaMA) drive natural language understanding to powerful levels, however, they still rely on historical, static, unstructured training data; in addition, LLMs have limited context windows of usage, and therefore are not functional in the real-time enterprise world. Retrieval-augmented generation (RAG) frameworks have arisen to solve the aforementioned issues by embedding real-time organizational knowledge into a vector-based semantic retrieval combined with a generative model to ground its output in relevant data. In addition to dynamic data usage, assistants will need to be multimodal (text, images, and audio, etc.), and multilingual. While there has been a range of recent work that combine these features (e.g., OpenFlamingo, Kosmos, mBERT, and XLM-R), there remain substantial engineering challenges related to embedding alignment, video/image/text/audio input fusion, prompt translation, insertion and prominence, contextual coherence, and other challenges. Finally, an enterprise-grade solution will need to be modular, secure, and scored; our system is built using microservices and includes pipelines for document file ingestion, a parser, embedding, semantic retrieval, and generative model output generation. It also includes role-based access control (RBAC).

II. RELATED SYSTEM

A. Kyndi

Kyndi is a cognitive search platform aimed at enterprises, offering natural language-based search over structured and unstructured data. It focuses on explainability, traceability,

and regulatory compliance key concerns for industries such as finance and legal. However, Kyndi's capabilities are mostly restricted to textual data and do not extend to multimodal or multilingual interaction. In contrast, our proposed RAG system supports a variety of data types including audio, image, and video, and it leverages large language models for dynamic generation of responses beyond pure search capabilities

B. IBM Watson Discovery

IBM Watson Discovery offers AI-enabled document retrieval and analytics across enterprise data silos. It combines natural language processing with enterprise search, enabling better insights from business documents. While Watson Discovery provides excellent integration with existing IT systems and strong search capabilities, it does not possess any flexible, open-ended generative capabilities and the ability to support long-context queries. In our RAG system, we use multimodal documents embedded in vector space, and generate contextualized answers that are based on a full document understanding.

C. Azure OpenAI on Enterprise Data

The Azure OpenAI service from Microsoft provides organizations with the ability to connect real GPT models with internal enterprise data sources through Azure Cognitive Search as a tool for retrieval-augmented chat over enterprise knowledge bases. However, deploying it involves considerable configuration, and built-in modalities for ingestion (e.g., parsing visual documents, ingesting voice notes) are not really supported. Our proposed system is multimodal and can be deployed on-prem, or in hybrid environments allowing organizations to better align with their data governance, compliance, and operational restrictions in sensitive sectors such as healthcare and transportation.

D. Glean

Glean is an enterprise search solution that is AI-powered, where information can be searched and discovered among applications such as Google Workspace, Slack, and Jira. Glean allows users to discover contextually-relevant information as needed. It does not support multimodal actions based on generative-AI capabilities. Glean is a good solution for app-level integrations, but our RAG assistant is built to provide a deeper level of semantic understanding, scalable retrieval, and generative summarization of multimodal content for more specialized use cases with complex documentation such as medical documentation, regulatory protocols and more.

III. PROPOSED SYSTEM

The proposed solution is a scalable, multimodal, and multilingual enterprise assistant based on a Retrieval-Augmented Generation (RAG) architecture. This system addresses key limitations of traditional enterprise knowledge platforms, particularly their inability to process unstructured and multimodal data in real time, and their lack of support for semantically enriched, user-specific responses. At its core, the system integrates a pre-trained Large Language Model (LLM) with a vector-based semantic retrieval engine, specifically Qdrant, enabling efficient and contextually relevant retrieval across a heterogeneous enterprise corpus. This design is informed by unified evaluations and best practices in retrieval consistency and factual accuracy as examined in [6], [4].

To support complex and lengthy documents such as policy manuals, medical histories, and technical specifications, the system incorporates LongRAG—a long-context variant of RAG that utilizes transformers capable of handling extended token sequences [9], [10]. This enables coherent multi-turn dialogue and accurate summarization across long inputs, outperforming standard RAG baselines in enterprise-scale reasoning tasks as shown in [5]. The architecture also employs a modular, agent-based framework, aligning with the agentic decomposition approaches in [2] and [3], wherein specialized agents independently handle context retrieval, evidence ranking, reasoning, and generation. This not only improves response quality but also enhances computational efficiency through parallel execution.

The assistant accepts multimodal input types including natural language text, OCR-processed scanned content, annotated visuals, and audiovisual data. Such capability is in line with techniques explored in [6] and is foundational to enabling interaction across varied enterprise workflows. Furthermore, the system supports multilingual understanding using models such as XLM-R and mBERT, allowing for semantic fidelity across multiple languages and dialects—an essential feature for international enterprise settings [5].

Domain-specific adaptability is a salient feature of the system. It supports fine-tuning on specialized corpora, such as those in healthcare and transportation, to optimize the contextual relevance and accuracy of outputs. This is informed by approaches in [7], which demonstrate Pareto-optimized LLM adaptation via contextual retrieval. To support operational scalability, the system is implemented using modular microservices and is deployable across on-premise, cloud, or hybrid infrastructures, in accordance with design principles emphasized in [4].

To enable continuous learning, the assistant includes a feedback loop that uses reinforcement learning to improve retrieval and generation strategies over time [1], [3]. This allows the model to dynamically refine its responses based on interaction history and real-world usage. In addition, the system’s architecture supports real-time operation through optimized pipelines and parallelizable modules, making it suitable for high-throughput enterprise environments [4].

IV. METHODOLOGY

The proposed private enterprise assistant system is evolved in a modular and structured fashion, integrating Retrieval-Augmented Generation (RAG) capabilities with secure role-based access control, and multimodal processing. In accordance with this methodology, the focus is on four main aspects: user access control, multimodal file ingestion and chunking, intelligent retrieval and generation; and practical applications in real-world enterprise use cases.

A. *System Roles and Access Management*

To accomplish timely, secure and coherent access to the system, a Role-Based Access Control (RBAC) mechanism is accomplished. The configuration officially states four distinct, unique and separate roles, Superadmin, Admin, Editor, and User, thus establishing clear delineation of authority and boundaries with responsibilities. The Superadmin has full privileges, including the ability to manage API keys, manage the global settings for the system, and manage and coordinate organizational workspaces. The Admin is charged with managing the uploading of files and managing individual projects within each workspace, with the participatory oversight of an Editor(s) and User(s) which they manage. The Editor is limited to either upload, organize and annotate documents, but have no privileges to management settings for the overall system or API integrations. The User will have the lowest-level access which allows him to only interact with the system by querying it through natural language request. The hierarchical approach ensures that data protection and security, as well as discipline in operation with processing activities across the platform, are robust and implemented.

B. *Multimodal File Ingestion and Preprocessing*

The system can accept many formats/documents, including goodly host of different document types (PDF, DOCX, XLSX, PPTX, TXT), images (JPG, PNG), audio files (MP3, WAV), and video(s) (MP4). When a file is uploaded, the first step is to check the file type and conduct a security scan. Text files are parsed by trusted libraries: pdfplumber, and PyMuPDF and structured data (tables, headers) is preserved when extracted. Images, and scanned PDFs are converted to text using Optical Character Recognition (OCR) tools: Tesseract, PaddleOCR. An additional issue exists with audio, and video files, the text of these files won't be extracted until they are transcribed. The speech-to-text applications can be used for transcription, for example OpenAI Whisper. Images will also need to have some sort of description created from them, this could a scene description or captions, this might require the use of a vision-language model, for example BLIP or CLIP. The outputs of this multimodal preprocessing stage ensure that all relevant points of data have been obtained, and prepared for semantic processing.

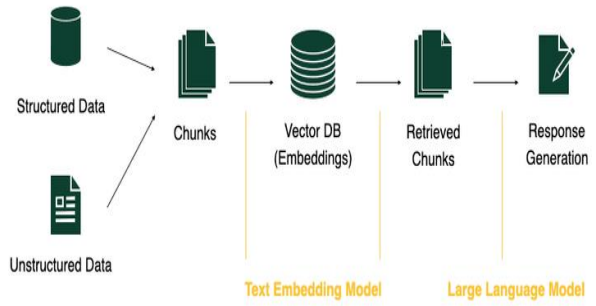

Fig 4.1 Architecture of RAG

The RAG architecture figure 4.1 combines a retriever and a generator to produce accurate, context-aware responses. A user query is first converted into a vector using a transformer encoder. This vector is matched against a pre-indexed corpus in a vector database to retrieve the top-k relevant documents. These documents, along with the original query, are passed to a generator model, which synthesizes a coherent, evidence-based answer. By retrieving real-time external knowledge during inference, RAG enhances the factual grounding and relevance of language model outputs, making it ideal for knowledge-intensive applications.

C. *Multilingual and Multimodal Support*

This platform is intended to accommodate both multilingual and multimodal user interactions. Users can upload content in various languages or query the assistant in multiple languages (e.g. English, Spanish, Arabic, Hindi), based on multilingual tokenizers and embedding models. Voice queries are integrated through real-time transcription with speech-to-text systems, in a similar manner as text-based queries. Images are processed using vision-language models to allow the user to ask questions related to visual aspects of the content. The multimodal and multilingual functionality of the assistant enables use by organizations and teams with geographically diverse locations and varying language preferences alongside a variety of technical backgrounds.

D. *Real-World Applications Across Domains*

This system is domain-agnostic, which allows for a deployment in multiple industries including, but not limited to, healthcare, transportation, legal, and education. A healthcare example: administrators upload patient records, patient reports, clinical documents, etc. Doctors then can query the documents in order to obtain certain medical histories or diagnostic guidelines. A railroad example: the system ingests regulatory documents, maintenance procedures, operational manuals, etc. And, railroad workers can query the documents to retrieve procedural steps, safety, or policy updates. In a law firm, administrators upload case files and legal references, and any personnel can retrieve precedents or compliance guidelines as needed. The versatility of the RAG model and the power of the file ingestion pipeline give us confidence that the system is suitable for the enterprise in a variety of domains.

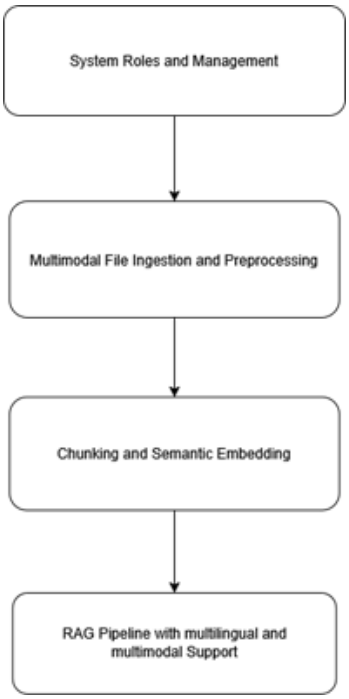

Fig 4.2 RAG System Feature Workflow

Figure 4.2 illustrates the end-to-end feature flow of the RAG system, highlighting the interaction between the retriever and generator components. It demonstrates how user queries are processed, relevant documents retrieved, and context-aware responses generated.

V. SYSTEM ARCHITECTURE

A. User Interaction Layer

The first layer of the whole system is the User Interaction Layer, which can be accessed from four different roles: superadmin, admin, editor and user. This layer is responsible for any user-related front-end interfaces that control the UI/UX, and provides for a natural approach for interaction with the assistant through text, images, and voice. The user layer is built with intended support of web frameworks and technologies, WebAssembly as well as web-based technologies MudBlazor, to allow for responsive capabilities with detection of context related to all types of users (superadmin, admin, user, editor). The interaction built into the user layer allows for multimodal invocation, i.e., document upload, speech recognition, text input chat-style interaction. For non-technical users, this does not stop at their use of the system but includes insights for documents, image generation if necessary, and completing the answers to users from documents they provided questions for, and without needing expertise.

B. Role-Based Access Control Layer

This layer implements the hierarchical access policies, dictating what a user can see and do depending on his/her role. For instance, a Superadmin has full privilege at the system level, giving the user capability over system configuration, management of API keys, and auditing system usage. Admins are permitted to configure organizational projects, upload and process documents, and manage editor-level entitlements. An editor is provided upload access and the ability to submit data but with no editorial capabilities or the ability to modify the configuration of the system as defined for their organization. In another development role, the user can interrogate the system but only regarding previously submitted queries and will have no rights to upload or to modify any configuration. Access logic is ingrained in the middleware and into the authentication constraint to ensure compliance with formal and informal policy rules throughout the entire system on behalf of organizations regarding permitted security and workflow rules.

C. Multimodal Input Processing Layer

Upon uploading the data, the Multimodal Input Processing Layer identifies, preprocesses, and structures the input. If the input is a PDF, DOCX, PPTX, MP4 or an audio recording, then the system sends the data, depending on the type of data being processed, to dedicated file conversion microservices (using LibreOffice, ffmpeg, and speech-to-text models) that will normalize the data into text and images. Then, depending on the type of data that is being processed, this layer will use OCR and image extraction tools to documents, transcribe the speech data into text, and extract frames from the video to be included. All of the structured outputs are sent to the next layer to be enriched and embedded.

D. Chunking and Embedding Layer

At the base of the assistant's retrieval system is the Chunking and Embedding Layer. In this layer, the pre-processed content is chunked into contextual text segments or "chunks" using adaptive chunking algorithms that maintain semantic context. The chunks are then embedded into high-dimensional vector representations with transformer-based encoders optimized for multi-lingual representation. The input is generated page-wise for text and, at the same time, image embeddings with captioned descriptions. Image embeddings by themselves allow the system to archive and retrieve data not only by textual context, but through visual semantics as well. The chunked embeddings are then individually indexed and stored in a vector database (Qdrant) for nearest neighbor query execution for similarity searches that require computational time-based contextual processing.

E. Metadata & Summary Generation Layer

To optimize the retrieval performance, a dedicated metadata generation layer is created per document. This layer generates summaries, keywords, tags, and document-level embeddings that serve as high-level semantic descriptions. These metadata vectors are stored alongside the chunked data in the vector store and allow for both fine-grained and coarse-grained lookup. The layer produces multilingual as well, so that it is available to users who are working in a language other than English.

F. Retrieval and RAG Model Execution Layer

This layer consists of the retrieval and generation components that form the basis of the system's intelligence and reasoning process. Upon receiving a user query, the retrieval engine will contact the Qdrant retrieval engine to acquire the most current, relevant chunks based on semantic similarity. The top-k retrieved contexts are passed to a fine-tuned generative language model that can perform context aware reasoning and answer generation. The language model can produce both extractive and generative responses while maintaining fact integrity and coherence within the responses. The model also demonstrates self-reasoning capabilities (as outlined by Yuan Xia et al. [1]) and can be adapted to generate agentic workflows [2] whereby, subsequent retrievals and reasoning responses are obtained to iteratively refine the response.

G. Auxiliary Intelligence Services Layer

This layer encompasses optional services which augment the assistant's overall capabilities. These being - Web Search Integration for real-time augmentation when internal data is not available; Image Generation and Search leveraging diffusion models and CLIP-based retrieval, respectively; and multilingual translation and transcription services for global organizational implementations. These services are modular and containerized, and can be turned on or off depending upon the project context.

H. Storage, Logging and API Layer

The lowest layer manages data persistence and auditability of the system. It is comprised of three primary stores: Relational Database (PostgreSQL/MySQL) for user & project metadata NoSQL Document Store (MongoDB) for document snapshots and intermediate content Vector

Database (Qdrant) for high-dimensional embeddings and indexing support for retrieval. All user upload, access, and query activity log-structured logs are generated. These are important for debugging, enterprise compliance, and auditing. API Gateway and Microservices Orchestration Layer. All functionality is made available via well-described APIs orchestrated via FastAPI based microservices. Each module is decoupled, containerized, and orchestrated via Docker or Kubernetes for horizontal scalability and resilience. Background task handling utilized for cleanup, progress or retry mechanisms are incorporated using Celery or similar job queues.

VI. FUTURE ENHANCEMENTS

A. Advanced Multimodal Support

One of the key developments for RAG-based systems, particularly in healthcare and industrial contexts, is the expansion of modality processing and integration capabilities. RAG currently uses audio, video, text, and images, and will soon add the ability to extend support to more complicated multimodal inputs like 3D scanning, interactive visualization, or augmented (AR) or virtual reality (VR) environments. An example of this is illustrated by Riedler and Langer's work with multimodal RAG: Using visual and textual data to obtain answers, attained improved AI-generated women-engaged product in an industrial context, then only using one data modality. Our system will implement Riedler and Langer's work by expanding system capabilities to extract context from complex image data, like MRIs, x-rays, and various diagnostic scans, while also making the system's visual data inputs more interactive for the user. These future expansions will allow the medical professional to not only query textual data but also visually query and disengage with visual data, to make better-informed decisions.

B. Improved Contextual Understanding with Long-Context Models

As Wang et al. have suggested in their work on long context question answering, many healthcare documents, for instance patient histories, lab reports, or discharge summaries, can be many dozens of pages long. Traditional language models have limited capacity for considering long documents and consequently lose key contextual information which may help formulate answers. When the application continues to be developed, we will use long context models like Llama3-ChatQA-2, which can process up to 128K tokens, and therefore the application will be able to handle and answer queries that require deep contextual understanding of large documents. Long context models used alongside RAG will allow users of the application to question entire patient histories and huge repository rulebooks in management of railways without losing relevant contextual materials

C. Personalized Patient and Worker Profiles

Another area of improvement is personalizing the assistant for individual healthcare workers or railway workers. This would allow the assistant to tailor its responses and suggestions based on user query history and usage patterns. For example, physicians may receive responses more specific to their area of specialty while other railway workers may receive only relevant safety policies and protocols based upon their department and responsibilities. This could be done with machine learning techniques, using predictive modeling for behavior and context-aware personalization not unlike Joshi has described regarding AI in healthcare, where the analysis of a user's previous queries are utilized to support the development of new suggestions. While this would also improve the relevancy of responses, it would also allow the assistant to be proactive in offering recommended responses or reminders in response to users requests

D. Real-Time Collaboration and Knowledge Sharing

The ability to perform real-time collaboration is a critical competency in professional work environments, particularly in situations such as healthcare where rapid decision-making can save lives. Later versions of the system will support real-time collaboration where health-care providers (for example, doctors, nurses, or medical research staff) and rail workers can collaborate, share insights, discuss results, and develop collaboration decisions. Some examples of real-time collaborative features that can be incorporated include live document annotations, chat systems, and collaborative workflows. Moreover, based upon Joshi's research on AI-based assistants in healthcare, by developing better workflows that allow collaboration it will improve quality of care and result in faster, more informed joint decisions.

E. AI-Driven Predictive Analytics

At last, predictive analytics will serve as an upgrade for future iterations of the system. The assistant will utilize machine learning algorithms to analyze historical data in order to anticipate future trends, such as the probability of certain diseases and medical outcomes associated with previous patient history, and inventory forecasting maintenance required in railway scenarios. For health information technology applications, Joshi's prioritization of predictive analytics can closely relate in constructing early warning systems for risks to the care of patients through preventative care more efficiently. In a similar regard, identifying maintenance required to mitigate systematic failure in railways, predictive analytics can anticipate system failure or maintenance so that maintenance occurs timely, and system downtime is minimized.

VII. CONCLUSION

This paper has discussed a retrieval-augmented-generation (RAG) solution for an enterprise level implementation intended to re-engineer knowledge-based workflows in mission-critical verticals (e.g., railways, healthcare). With this solution, advanced language models work synergistically with semantic search technology like Qdrant to ensure that an enterprise user can interrogate relatively complex multimodal content (i.e., streaming text, image, audio, video) using natural language. The platform has been custom-designed to offer important aspects of enterprise-level data governance: scalability/distribution, secured access, multilingual access, and role-based access. The platform's modular architecture can be deployed on-premise, cloud-based or hybrid, and the local data governance realities and regulatory constraints can be adhered to with

respect to how data jurisdiction is enacted. It also provides significant enhancements to traditional RAG architectures through the ability to conduct multimodal retrievals, achieve long-context understanding, ingest domain-specific pre-trained or real-time knowledge, and provide conversational-user-interfaces. Most importantly, while previous innovation in academic literature has inspired the way this RAG concept has been developed (i.e., Riedler & Langer; Wang et al.), it operationalizes generative AI going beyond conceptual prototype development to the trial operationalization of generative AI in real enterprise contexts. This technological platform enables swift access to high-value and time-sensitive information. It is designed to lessen cognitive load by streamlining an enterprise collaboration engagement for users in various roles through effective multitasking and makes many relatively low-value, time-sensitive decision-making easier to ensure better outcomes (i.e., at the observatory stage - not deciding to trade-off cognitive effort, e.g., diagnostic information, with time). This can represent significant enhancements in mission-critical decision-making, e.g., communications protocols around rail safety (accident/incident), rail medical diagnostics (both CPR decisions and not CPR decisions), etc. Over the next 12-24 months, the technical architecture will continue to evolve into a full-scale comprehensive helper or assistant for knowledge base industries with even more add-on enhancements such as predictive analytic capabilities, enhanced basic domain adaptation, accurate collaborative analysis and capability, and personalized AI agents with personalized avatars.

REFERENCES

[1] “Improving Retrieval Augmented Language Model with Self-Reasoning” paper by Yuan Xia, Jingbo Zhou, Zhenhui Shi, Jun Chen, Haifeng Huang

[2] “Agentic Retrieval-Augmented Generation: A Survey On Agentic Rag” paper by Tala Talaei Khoei, Saket Kumar, Aditi Singh

[3] “Improving Retrieval-Augmented Generation through Multi-Agent Reinforcement Learning” paper by Yiqun Chen, Lingyong Yan, Weiwei Sun, Xinyu Ma, Yi Zhang, Shuaiqiang Wang, Dawei Yin, Yiming Yang, Jiaxin Mao

[4] “Enhancing Retrieval-Augmented Generation: A Study of Best Practices” paper by Siran Li, Linus Stenzel, Carsten Eickhoff, Seyed Ali Bahrainian

[5] “Long Context vs. RAG for LLMs: An Evaluation and Revisits” paper by Xinze Li, Yixin Cao, Yubo Ma, Aixin Sun

[6] “Fact, Fetch, and Reason: A Unified Evaluation of Retrieval-Augmented Generation” paper by Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, Manaal Faruqui

[7] “Pareto-Optimized Open-Source LLMs for Healthcare via Context Retrieval” paper by Jordi Bayarri-Planas, Ashwin Kumar Gururajan, Dario Garcia-Gasulla

[8] “RankRAG: Unifying Context Ranking with Retrieval-Augmented Generation in LLMs” paper by Yue Yu, Wei Ping, Zihan Liu, Boxin Wang, Jiaxuan You, Chao Zhang, Mohammad Shoeybi, Bryan Catanzaro

[9] “LongRAG: Enhancing Retrieval-Augmented Generation with Long-context LLMs” paper by Ziyan Jiang, Xueguang Ma, Wenhu Chen

[10] “Retrieval meets Long Context Large Language Models” paper by Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, Bryan Catanzar