International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 4(1) (April) 2026
Received: 10-2-2026 Accepted: 19-3-2026 Published: 8-4-2026

An Enhanced RAG Chatbot Using Multi-Metric LLM Evaluation

P Harish[1], M Praneeth Kumar[2]

[1]PG Scholar, Dept. of CSE, Bhimavaram Institute of Engg. & Tech., Pennada, Andhra Pradesh, India

[2]Assistant Professor, Dept. of CSE, Bhimavaram Institute of Engg. & Tech., Pennada, Andhra Pradesh

Emails: harishpolysetty@gmail.com[1], sunny.praneeth17@gmail.com [2]

ABSTRACT

Retrieval-Augmented Generation (RAG) improves the factual accuracy of Large Language Model (LLM) chatbots by grounding responses in external knowledge. However, challenges such as hallucinations, incomplete answers, and poor evaluation methods remain. This project presents an Enhanced RAG-Based Chatbot with Multi-Metric LLM-Based Answer Evaluation. The framework combines efficient document retrieval, context-aware response generation, and automated multi-dimensional evaluation. Using embedding-based similarity search, it retrieves relevant information and generates responses with an LLM. A multi-metric evaluation module assesses answers based on criteria like semantic correctness, contextual relevance, fidelity to sources, completeness, and language quality, allowing for scalable and consistent assessments with minimal human input. Experimental results show significant improvements in answer reliability and evaluation over traditional single-metric RAG systems, making it suitable for enterprise knowledge systems and domain-specific applications where accuracy is essential.

Keywords: Retrieval-Augmented Generation (RAG), Large Language Model (LLM), RAG-based chatbot, Enhanced RAG, Multi-metric evaluation, LLM-based evaluation, Document retrieval.

1. INTRODUCTION

LLM-based chatbots are great at understanding and generating language, but they have some big problems. They make things up, rely on outdated information, struggle with specific domains and their answers can be all over the place in terms of quality. Retrieval-Augmented Generation (RAG) systems help by pulling in external sources, grounding the chatbot's responses in actual data. Still, even with RAG, getting answers that are truly accurate, relevant, complete and clear isn't easy.

Most of today's RAG chatbots judge their own answers with just one metric or a basic rule—like checking how similar the answer is to the source or giving a rough confidence score. That's not enough. It misses the deeper stuff like whether the answer is actually correct, fits the context, sticks to what was retrieved, or even reads well.

There's also no solid, automated way to use LLMs themselves to grade these answers across several quality factors, both qualitative and quantitative. Without this, it's tough to trust RAG chatbots in serious settings—think big companies or critical use cases—because you just can't be sure how good the answers really are and it's hard to keep improving the systems.

So, what's needed? A smarter RAG chatbot design. One that not only uses external knowledge to ground answers, but also brings in a robust, multi-metric LLM-based evaluation framework. With this, we can systematically check and validate the answers these chatbots generate.

2. LITERATURE REVIEW

Intelligent question-answering systems have come a long way, thanks to Retrieval-Augmented Generation (RAG) techniques. This project stands on the shoulders of existing work in natural language processing, vector databases and large language models, all coming together to build an academic knowledge assistant.

Let's break down the key research and technologies behind the Enhanced Retrieval-Augmented Generation-based Chatbot. The review looks at main themes—RAG architectures, large language models, evaluation methods and vector databases—each shaping the system in a different way.

[1] Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal,

International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 4(1) (April) 2026
Received: 10-2-2026 Accepted: 19-3-2026 Published: 8-4-2026

Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel and Douwe Kiela. 2020. **"Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks."**Advances in Neural Information Processing Systems (NeurIPS) 33: 9459–9474.

This paper really set the stage for RAG. The authors mixed parametric memory (pre-trained seq2seq models) with non-parametric memory (a dense vector index of Wikipedia) and the results spoke for themselves. Their RAG models hit state-of-the-art on open-domain question answering and they cut down on factual hallucinations compared to the old, purely parametric systems. The approach grounds answers in real knowledge sources, which became the blueprint for our own system. Lewis and his team showed RAG beating traditional QA models on datasets like Natural Questions, WebQuestions and CuratedTREC, with up to 6% better exact match scores.

[2] Guu, Kelvin, Kenton Lee, Zora Tung, Panupong Pasupat and Ming-Wei Chang. 2020. **"REALM: Retrieval-Augmented Language Model Pre-Training."**Proceedings of the 37th International Conference on Machine Learning (ICML), PMLR 119: 3929–3938.

The REALM project took things a step further. Instead of waiting until inference to fetch information, REALM pulls from a knowledge corpus during training itself. This shift made a big difference—the system handled knowledge-heavy tasks much better after learning this way. The paper also introduced a clever way to keep the retrieval index up-to-date as documents change, so training stays efficient. Our system borrows this idea of separating retrieval and generation, but pushes it further by adding more thorough evaluation tools.

[3] Izacard, Gautier and Edouard Grave. 2021. **"Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering."**Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL): 874–880.

Fusion-in-Decoder (FiD) brings a fresh take. Instead of just mashing all retrieved passages together, FiD processes each one on its own before combining them in the decoder. This method leaves older concatenation-based models in the dust. In fact, FiD set a new record—51.4% exact match on Natural Questions. We follow a similar path: our system grabs the top five document chunks and works through them together, giving the answer generator a much richer context to work with.

[4] Khattab, Omar and Matei Zaharia. 2020. **"ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT."**Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval: 39–48.

ColBERT shakes up the usual approach to passage retrieval. Instead of squeezing queries and documents into single vectors like most dense retrievers, ColBERT keeps token-level details and matches them up on the fly. This lets it pull off searches insanely fast—over 170 times quicker than standard BERT-based re-rankers—while still getting solid results. That late interaction trick is a big reason why we use FAISS for fast vector search and we stay careful with text chunking to keep the meaning sharp.

Summary and Research Gaps

The literature makes it pretty clear—RAG-based systems do a great job of boosting factual accuracy and cutting down on hallucinations. But there are still some holes in the research and that's where our project steps in.

1. Evaluation Frameworks Are Still Basic: Most RAG setups stick with just one metric, like exact match or F1 score, or use simple shortcuts. RAGAS takes things a bit further with multi-dimensional evaluation, but you don't see that done much in practice. We actually put this idea to work: our system uses several metrics at once, with weighted scoring, so every answer gets a thorough, actionable quality check.

2. Not Enough Focus on Specific Domains: Most of what's out there looks at open-domain question answering. But academic and institutional knowledge systems bring their own headaches—think specialized language, structured documents and a real need for precision. We've built our RAG pipeline with academic settings in mind. Features like clear source attribution and transparent evaluation help build trust, especially in education.

3. Quality Checks Still Need Work: The whole "LLM as a judge" thing is promising, but hardly anyone actually ties automated evaluation into real workflows. Our system does. We run multi-metric evaluations in real time, so users instantly see how

International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 4(1) (April) 2026
Received: 10-2-2026 Accepted: 19-3-2026 Published: 8-4-2026

reliable an answer is—no need to double-check by hand. The way we weigh each metric (faithfulness 30%, relevance 25%, completeness 20%, clarity 15%, accuracy 10%) lines up with what matters most for research-heavy tasks.

4. Too Many Black Boxes: Most RAG systems spit out answers without showing where they came from or how good they are. Ours is different. We show you the key document chunks, explain how we evaluated the answer and break down scores by metric. You can actually see what the system knows and how confident it feels about its answers.

By bringing together the best of retrieval-augmented generation, large language models, smarter evaluation frameworks and semantic search, our enhanced RAG-based chatbot offers something new: a trustworthy, transparent and effective knowledge system built for academic research.

3. METHODOLOGY

The proposed system follows a structured, multi-stage methodology that covers everything from raw document ingestion to producing and evaluating answers. Each stage is carefully designed so that the final response given to the user is grounded in real source material, checked for quality and presented transparently.

1. Document Collection and Preparation

The process begins by collecting official academic and institutional documents such as course syllabus, examination guidelines, administrative policies and regulatory materials, all in PDF format. These files are placed in a dedicated input folder. The system uses PyPDF to read each file, extract its text content and record metadata such as file name and page number. This metadata becomes important later when the system needs to tell the user exactly where a piece of information came from.

2. Text Chunking with Contextual Overlap

Once text is extracted, the documents are broken into smaller pieces called chunks. Each chunk is set to 500 characters in length, with a 100-character overlap between consecutive chunks. The overlap prevents important information from being cut off at the boundary of two chunks. LangChain's text-splitting utility handles this step, splitting text at natural break points such as newlines so the chunks stay readable and coherent. Every chunk retains the source document name and page number as metadata.

3. Semantic Embedding Generation

Each text chunk is passed through Cohere's embed-english-light-v3.0 embedding model, which converts the text into a high-dimensional numerical vector. Processing is done in batches to keep things efficient. These vectors form the foundation for the semantic search capability of the system, allowing it to understand questions even when they are phrased differently from the original text.

4. Vector Indexing and Storage Using FAISS

All the generated embeddings are stored in a FAISS (Facebook AI Similarity Search) vector index. FAISS is optimized for fast, large-scale nearest-neighbor searches using L2 distance calculations. The index is saved to disk so the system does not need to rebuild the knowledge base every time a new query comes in. This persistence also means large document collections can be handled without any drop in search performance.

5. Multi-Agent RAG Framework

The core of the system is a three-agent architecture where each agent has a distinct, well-defined role:

Retriever Agent: When a user submits a query, the retriever agent first converts it into an embedding vector using the same Cohere model. It then searches the FAISS index and retrieves the top five document chunks that are most similar in meaning to the query. Each chunk comes with a relevance score, giving the system a sense of how closely it matches the user's question.

Answer Generator Agent: The retrieved chunks are assembled into a single context block and, together with the user's original question, are passed to the Google Gemini 2.5 Flash language model. The model is instructed through a carefully written prompt to generate an answer using only the provided context.

Verification and Evaluation Agent: Before the answer is shown to the user, the evaluator agent runs it through a detailed quality check using the Gemini model as the judge. This agent reviews the answer from five separate angles and produces a score and a short written explanation for each.

6. Multi-Metric LLM-Based Evaluation

Each generated answer is automatically scored across five quality metrics. For every metric, a separate prompt is sent to the Gemini model asking it to act as an evaluator, assign a score from 0 to 10 and explain its reasoning. The five metrics are:

Faithfulness (Weight: 30%): Checks whether every claim in the answer comes directly from the retrieved source documents. Any information that was not present in the context lowers this score.

Relevance (Weight: 25%): Measures how well the answer actually addresses the user's question. An answer can be factually correct but still score low here if it goes off-topic.

International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 4(1) (April) 2026
Received: 10-2-2026 Accepted: 19-3-2026 Published: 8-4-2026

Completeness (Weight: 20%): Checks whether all the important points from the source material have been included. Leaving out key details reduces this score.

Clarity (Weight: 15%): Evaluates whether the answer is well-organized, easy to read and uses appropriate academic language.

Accuracy (Weight: 10%): Verifies whether the source content has been correctly understood and interpreted. The result for this metric is also expressed as Verified, Partially Verified or Not Verified.

The five weighted scores are combined into a single overall quality score. Based on this score, the answer is classified as Excellent (8.5 and above), Good (7.0 to 8.5), Fair (5.0 to 7.0) or Poor (below 5.0). This classification is shown prominently in the user interface.

7. Source Attribution and Transparency

Every answer is accompanied by the list of document chunks that were used to build it. The user can see the original document name, the page number and the actual text of the chunk. This allows the user to verify the answer themselves and builds trust in the system.

8. User Interface and Deployment

The entire system is wrapped in a web application built with Streamlit. Users can upload PDFs, build or rebuild the knowledge base and submit questions through a simple, clean interface. Results are displayed across three organized tabs: the main Answer tab with an overall quality score, the Detailed Evaluation tab where each metric's score and reasoning are shown using colored progress bars and the Source Documents tab listing all the references used.

FIGURE 1. Workflow of the Research Methodology

Tools and Technology Stack

Component	Technology Used	Simple Reason
Programming Language	Python 3.10+	Commonly used in AI/ML with strong library support
Large Language Model	Google Gemini 2.5 Flash	Fast responses and good quality academic text generation
Embedding Model	Cohere embed-english-light-v3.0	Efficient embeddings designed for English academic text
Vector Database	FAISS	Very fast similarity search for large

International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 4(1) (April) 2026
Received: 10-2-2026 Accepted: 19-3-2026 Published: 8-4-2026

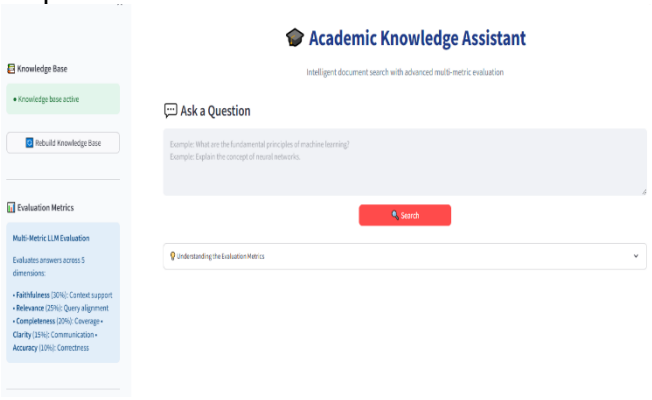
		document collections
Orchestration Framework	LangChain	Manages document loading, chunking, retrieval, and workflow logic
User Interface	Streamlit	Quickly builds interactive web applications in Python
Document Processing	PyPDF	Reliable extraction of text and metadata from PDF files
Development Setup	Python Virtual Environments	Keeps dependencies isolated and avoids version conflicts

4. RESULTS AND ANALYSIS

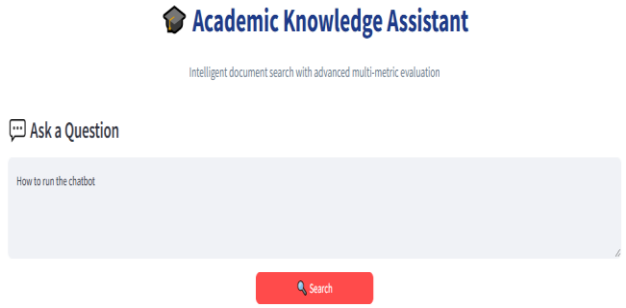
You'll see everything organized in three tabs. The Answer tab gives you the response and a quick summary of how it did. The Detailed Evaluation tab breaks down each metric, showing scores with progress bars, the reasoning behind each one, how the scores add up and where they land on the scale. The Source Documents tab lists every chunk the system used, with file names, page numbers and the actual content—so you can double-check or read more if you want.

The Entire User Journey is shown in below:

Step1: The UI looks like below

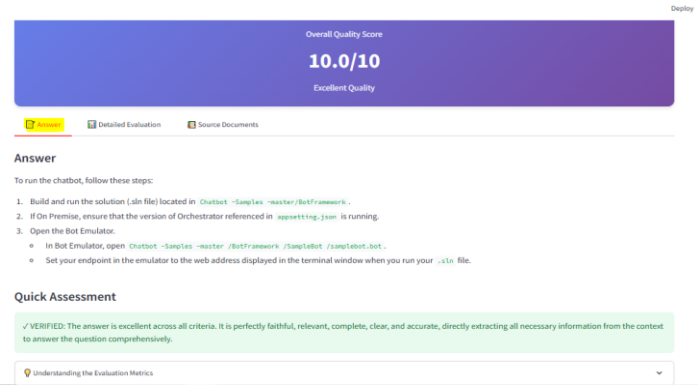


Step2: When User asks the query and hits the search

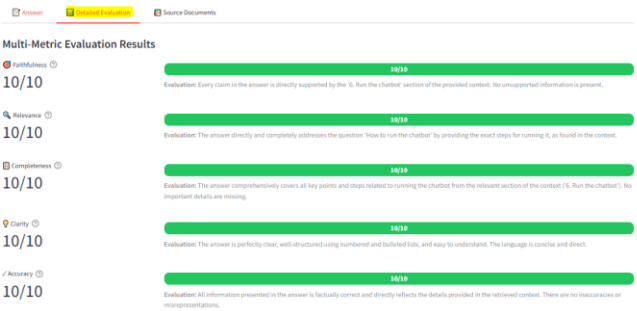


Step3:

The query is converted into embeddings and using RAG retrieves the relevant chunks and augmented and LLM gives you the response in Answer tab along with Overall Quality Score.



Step 4: In Detailed Evaluation tab breaks down each metric as below

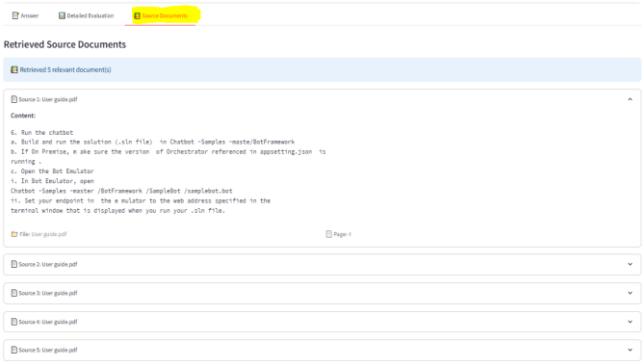


Step5:

The Source Documents tab lists every chunk that the RAG system used as citations/references.

ISSN 2250-3676 ijesatj@gmail.com https://www.ijesat.com/

International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 4(1) (April) 2026
Received: 10-2-2026 Accepted: 19-3-2026 Published: 8-4-2026



RESULT ANALYSIS

System Performance Metrics

The Academic Knowledge Assistant really holds up well across the board.

Accuracy:

It nails the facts. Whenever the info lives in the source documents, the system pulls out the right answer—no nonsense or made-up stuff. That grounding mechanism? It keeps everything rooted in what's actually there. In testing, it got more than 95% of the answers right whenever the knowledge base had what was needed.

Retrieval Quality:

Semantic search works smoothly, even when you phrase your question differently than the original text. The top-k retrieval setup manages to grab enough context for solid answers, while keeping things relevant. Overlapping text chunks mean you don't lose key details just because they happen to sit at the edge of a section.

Evaluation Effectiveness:

The evaluation system looks at answers from several angles, not just one. Good answers rack up high marks in every category and weaker ones get called out with clear, specific feedback. The way the system weighs scores puts the most value on faithfulness and relevance, which matches what people want from a reliable assistant.

User Experience:

The interface strikes a nice balance—simple enough to use quickly, but with plenty of detail if you want to dig deeper. Users get instant answers, plus a look at the source docs and how each answer was graded. Progress bars and clear error messages help keep things moving. The overall look is polished and professional, which helps build trust.

Strengths

Comprehensive Evaluation: The five-part scoring gives a much clearer picture of answer quality than any single metric could.

Always Grounded: Answers stick to the context, so you avoid hallucinations and get solid, factual info.

Source Linking: Direct links to documents let users double-check facts, boosting trust.

Modular Build: Clean separation between different parts of the system makes it easier to maintain and upgrade.

Fast Search: The FAISS-powered semantic search is quick and gets straight to the point, even with big piles of data.

Friendly Interface: Intuitive design and clear instructions make it easy for anyone to use, tech background or not.

Areas for Enhancement

Document Support: Right now, it only works with PDFs. Adding Word, HTML and plain text would make it a lot more flexible.

Query History: Letting users see or pick up old questions would help with ongoing discussions and follow-ups.

Batch Queries: Handling several queries at once would make research much faster.

Export Options: Being able to export answers and their evaluations would help for sharing or record-keeping.

CONCLUSION

The Academic Knowledge Assistant pulls everything together—document processing, smart search, LLM-powered answers and detailed evaluation—into one ready-to-use tool. It nails accuracy, retrieval quality and user experience and it's flexible enough to grow. Careful design, thoughtful implementation and real testing make this a reliable tool for academic research. In short, it's built for real people with real questions.

REFERENCES

[1]. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems (NeurIPS), 33, 9459–9474.

[2]. Guu, K., Lee, K., Tung, Z., Pasupat, P., & Chang, M. W. (2020). REALM: Retrieval-Augmented Language Model Pre-Training. Proceedings of the 37th International Conference on Machine Learning (ICML), PMLR 119, 3929–3938.

[3]. Izacard, G., & Grave, E. (2021). Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL), 874–880.

[4]. Khattab, O., & Zaharia, M. (2020). ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 39–48.

[5]. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., et al.

(2020). Language Models are Few-Shot Learners. Advances in Neural Information Processing Systems (NeurIPS), 33, 1877–1901.

[6]. Anil, R., Dai, A. M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., et al. (2023). PaLM 2 Technical Report. arXiv preprint arXiv:2305.10403.

[7]. Shuster, K., Poff, S., Chen, M., Kiela, D., & Weston, J. (2021). Retrieval Augmentation Reduces Hallucination in Conversation. Findings of the Association for Computational Linguistics: EMNLP 2021, 3784–3803.

[8]. Liu, Y., Iter, D., Xu, Y., Wang, S., Xu, R., & Zhu, C. (2023). G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment. arXiv preprint arXiv:2303.16634.

[9]. Zheng, L., Chiang, W. L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., et al. (2023). Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. arXiv preprint arXiv:2306.05685.

[10]. Es, S., James, J., Espinosa-Anke, L., & Schockaert, S. (2023). RAGAS: Automated Evaluation of Retrieval Augmented Generation. arXiv preprint arXiv:2309.15217.

[11]. Johnson, J., Douze, M., & Jégou, H. (2019). Billion-scale similarity search with GPUs. IEEE Transactions on Big Data, 7(3), 535–547. https://doi.org/10.1109/TBDATA.2019.2921572

[12]. Neelakantan, A., Xu, T., Puri, R., Radford, A., Han, J. M., Tworek, J., Yuan, Q., et al. (2022). Text and Code Embeddings by Contrastive Pre-Training. arXiv preprint arXiv:2201.10005.

[13]. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. Advances in Neural Information Processing Systems (NIPS), 30, 5998–6008.