Investigating Flavors of RAG for Applications in College Chatbots

Christian Sarmiento[a] and Eitel J. M. Lauría[b]
School of Computer Science and Mathematics, Marist University, Poughkeepsie, NY 12601, U.S.A.
{*christian.sarmiento1, eitel.lauria*}*@marist.edu*

Keywords: Retrieval Augmented Generation, Machine Learning, Large Language Models, AI, NLP, Higher Education.

Abstract: Retrieval Augmented Generation (RAG) has become a growing area of interest in machine learning (ML) and large language models (LLM) for its ability to improve reasoning by grounding responses in relevant contexts. This study analyzes two RAG architectures, RAG's original design and Corrective RAG. Through a detailed examination of these architectures, their components, and performance, this work underscores the need for robust metrics when assessing RAG architectures and highlights the importance of good quality context documents in building systems that can mitigate LLM limitations, providing valuable insight for academic institutions to design efficient and accurate question-answering systems tailored to institutional needs.

1 INTRODUCTION

The development of large language models (LLMs) has garnered significant interest across fields for their ability to generate content. Generative AI has impacted a large number of domains and applications, one of them being question-answering (QA). The ability of LLMs to generate accurate responses has sparked significant discussion, revolutionizing various fields while posing unprecedented adaptation challenges (Yunfan Gao, 2024). Particularly, LLMs have been an increasingly developing topic of conversation in academia due to their many benefits and challenges to learning and teaching (Crompton and Burke, 2023; Damiano et al., 2024). Some colleges have even gone as far as implementing AI in their administrative uses.

Although very useful, LLMs are still highly susceptible to hallucinations or generations that are not accurate or blatantly false or ludicrous, such as generating non-existent web links and references (Wu et al., 2024) or fabricating legal cases (Browning, 2024). Because of this, researchers in the question-answering domain have developed systems that, when presented with a query, can refer back to the corpus of text they were trained on and external knowledge to output (more) accurate answers (Gonzalez-Bonorino et al., 2022). LLM-based question-answering systems have experienced exponential growth, starting with the retriever-reader architecture, first proposed by researchers at Facebook AI (now Meta) and Stanford (Chen et al., 2017) and evolved into more sophisticated systems. The latest paradigm in developing LLM-based QA systems is called retrieval augmented generation (Lewis et al., 2020), also known by its acronym RAG. In this family of models, and as in previous retriever-reader architectures, the system uses the given query to find related documents; but instead of retrieving the best span of text from the retrieved documents, the GenAI component of the RAG mechanism generates the answers. The corpus of documents the retriever component uses can include resources such as internal policy handbooks, textbooks, or instructional guides tailored to address specific user queries. This helps mitigate the hallucinations an LLM may have when dealing with a user query (Yunfan Gao, 2024). Since the introduction of RAG in recent years, many other variations of RAG have been developed to deal with the issues that may arise with this paradigm in different application scenarios. This type of paradigm can be beneficial in a higher education setting, with multiple use cases and applications, including a) question-answering assistants (or chatbots) for administrative tasks or to answer frequently asked questions by students; b) AI-based teaching assistants delivered by instructors for their students -instructors can load lecture notes and solved exercises, and students can formulate questions to the TA to help enhance their learning process; c) AI-based teaching aids for instructors.

This study surveys RAG architectures and their application in higher education and analyzes two vari-

[a] https://orcid.org/0009-0009-4805-6904
[b] https://orcid.org/0000-0003-3079-3657

Sarmiento, C. and Lauría, E. J. M.
Investigating Flavors of RAG for Applications in College Chatbots.
DOI: 10.5220/0013468200003932

In *Proceedings of the 17th International Conference on Computer Supported Education (CSEDU 2025) - Volume 2*, pages 421-428
ISBN: 978-989-758-746-7; ISSN: 2184-5026

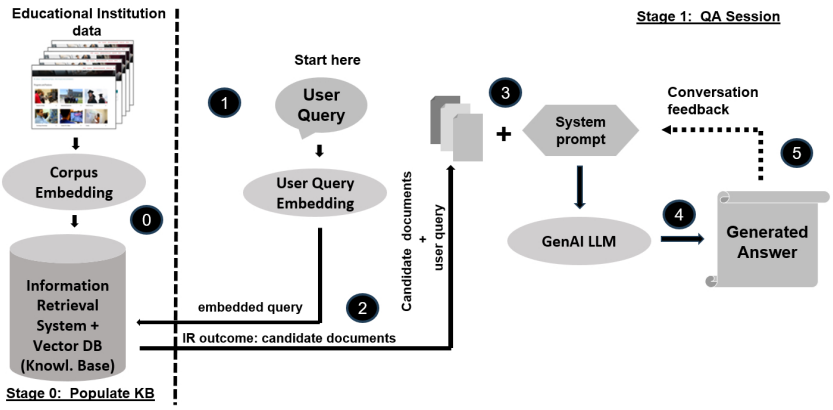

Figure 1: RAG Architecture.

ations: RAG's original implementation and Corrective RAG (Shi-Qi Yan, 2024), describing how they can be implemented and evaluated.

The paper is organized in the following manner: we start by introducing the retrieval augmented generation (RAG) architecture and the metrics used to evaluate its performance, followed by a short literature review of RAG technology in higher education. We then trace the most relevant variations of RAG technology developed since its inception, focusing on Corrective RAG. We follow with a detailed description of the experimental setup to analyze the performance of Simple RAG and Corrective RAG. We report and discuss our findings. Finally, we provide conclusions with comments on the limitations of the research.

2 RETRIEVAL AUGMENTED GENERATION (RAG)

2.1 RAG Architecture

RAG was first introduced as a paradigm for natural language processing tasks (NLP) in 2020 in a paper published by Facebook AI (now Meta) titled *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. The basic framework first involves embedding the corpus of context into a vector database. Then, using the vector database as a retrieval system, user queries are embedded into vector representations, which are then used to retrieve relevant documents; the LLM subsequently uses those documents to generate an answer to the query (Lewis et al., 2020). Figure 1 depicts the architecture and the flow of execution. In the document storage stage (labeled as 0), textual data extracted from the educational institution are transformed into vector embeddings and stored in a knowledge base (vector database) / information retrieval system for use by the question-answering conversational system. Then in the query stage: 1) The user submits a query; 2) The user query is vector-encoded and delivered to the information retrieval system where, through semantic / similarity search, a set of candidate documents is retrieved from the vector database; 3) Using the retrieved context and the user query, the LLM is then prompted, using engineered prompts, for a generation; 4) The generative LLM uses these data to produce the answer to the formulated user query; 5) If the conversation ensues, previous answers are added to subsequent tailored prompts to provide continuity to the flow of the conversation.

This approach mitigates hallucinations within LLMs since it combines the parametric memory of an LLM, or the internal knowledge it has acquired through training, with the retrieved context, which results in generations that are more grounded in context and thus more accurate and logically based (Lewis et al., 2020). Although RAG has proven to be very helpful for accurate generations in LLMs, there are still a number of issues that can make RAG non-reliable. In a simple RAG implementation, the effectiveness of RAG is highly dependent on the retrieval quality (Yu et al., 2024): retrieving documents that are inaccurate, contradictory, or unnecessary.

2.2 RAG in Higher Education

The inception of large language models, generative AI, and retrieval augmented generation technologies has bolstered the use of question-answering assistants in higher education. RAG, in particular, has become a promising architecture for faculty and administrators, given its potential for controlling hallucinations through the use of a curated corpus of data,

prompting, and logic workflow that constrains spurious generations and the natural ability of generative LLMs for producing cogent human-like text, in this case, summarized from retrieved documents. Several projects and publications have emerged recently in computer science education; see (Lyu et al., 2024; Wong, 2024; Thway et al., 2024) for example. OwlMentor (Thüs et al., 2024) is a RAG system designed to assist college students in comprehending scientific texts. (Modran et al., 2024) implement an intelligent tutoring system by combining a RAG-based approach with a custom LLM.

RAG Systems are considerably more robust when compared to answers delivered by LLMs, but can still exhibit hallucinatory behavior under certain circumstances. According to (Feldman et al., 2024), for example, RAG systems can be misled when prompts directly contradict the LLM's pre-trained understanding. This obviously has a significant impact in an educational setting where accuracy in answers is paramount. A critical outstanding issue to deploy at a large scale is credibility: in a study conducted by (Dakshit, 2024), a major recommendation was to have faculty members monitor usage to verify the correctness of the systems's responses. It is, therefore, imperative to objectively measure RAG systems' performance and the accuracy of the generations resulting from user queries.

2.3 Evaluation Metrics for RAG

Metrics are crucial for evaluating how well a RAG system is generating responses. Traditional statistical metrics alone fail to capture RAG processes' nuanced and dynamic nature. Evaluating a RAG system entails assessing the performance of its actions (retrieval and generation) and the overall system to capture the compounding effect of retrieval accuracy and generative quality (Barnett et al., 2024). Several benchmarks have been developed to measure the performance of RAG applications. In this work, we use *RAGAS: Automated Evaluation of Retrieval Augmented Generation* (Es et al., 2023). RAGAS is utilized in this study to analyze various subprocesses within the larger RAG framework, using several metrics: context recall, context precision, faithfulness, and semantic similarity.

Context recall focuses on how many relevant documents were successfully retrieved. If GT is ground truth and C is context, then context recall is calculated using this formula:

$$\text{Context Recall} = \frac{GT \text{ claims in } C}{\text{number of claims in } GT} \tag{1}$$

With context recall, a better sense of retrieval efficiency within a RAG process can be obtained by assessing RAG's ability to retrieve all necessary and related context for a given query. This measure is on a scale from 0 to 1, and high recall indicates that a significant proportion of relevant documents were successfully retrieved.

Context precision is defined by RAGAS as a metric that gauges the proportion of relevant chunks in the retrieved contexts. Mathematically:

$$\text{Context Precision} = \frac{\sum_{k=1}^{K}(\text{Precision@k} \times v_k)}{r_K} \tag{2}$$

with $v_k \in (0,1)$ equal to the relevance indicator at rank k, r_K equal to the total number of relevant items in the top K results, and Precision@k formulated using the equation below:

$$\text{Precision@k} = \frac{\text{TP@k}}{\text{TP@k} + \text{FN@k}} \tag{3}$$

'@k', a typical notation in information retrieval, indicates that the precision is computed only for the top k retrieved items rather than considering the entire set of retrieved contexts. TP@k and FP@k are the number of relevant and irrelevant chunks respectively, retrieved up to rank k. Context Precision uses an average of Precision@k values, weighted by v_k, which ensures that only relevant chunks contribute to the average. In that manner, retrieved chunks are evaluated based on their usefulness.

Faithfulness measures the factual consistency of the generation in relation to the retrieved context. A generation can be considered faithful if all the claims within the generation can be directly inferred from the retrieved context. A higher score, scaled from 0 to 1, indicates that everything claimed in the generation can be inferred from the retrieved context. If G is the generation and C is the retrieved context, then:

$$\text{Faithfulness} = \frac{\text{\# claims in } G \text{ supported by } C}{\text{Total \# of claims in } G} \tag{4}$$

This is a valuable metric for evaluation since it gives a compounded perspective of the retrieval accuracy and the generation quality in one cohesive metric.

Semantic similarity evaluates how semantically related the generation and the context documents are to one another. Semantic similarity is calculated by vectorizing the generation and context and using cosine similarity for the metric, which takes the cosine angle between the two vectors. This is on a scale from 0 to 1, with high scores indicating a high semantic similarity between the generation and contexts.

For the purpose of this study, RAGAS metrics are supplemented with an additional metric, labeled *Face Validity*, which is assessed through visual inspection of the experimental results to measure end-to-end the quality of the generation given the user query. For more details, see section 4.1.

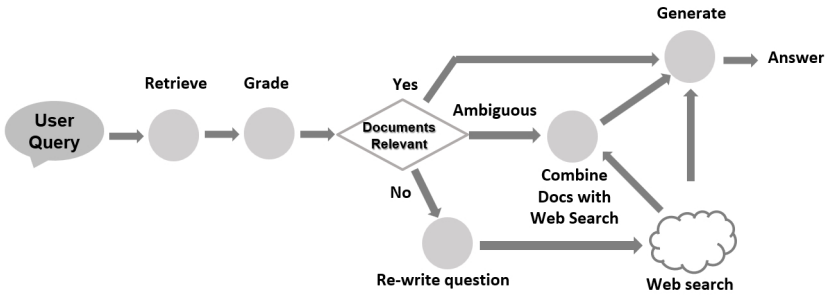

Figure 2: Corrective RAG Architecture - adapted from (Shi-Qi Yan, 2024).

3 RAG VARIATIONS

Since its inception in 2020, there has been plenty of research and development on the RAG architecture. RAG's original paper used dense passage retrieval (Karpukhin et al., 2020) to embed the documents in a dense, high-dimensional vector space. REALM (Guu et al., 2020) integrates LLM pre-training with the retriever, allowing the model to retrieve documents from the corpus of data used during pre-training or fine-tuning. dsRAG, an open-source retrieval RAG engine (D-Star-AI, 2025) implements Relevant Segment Extraction (RSE), a query-time post-processing algorithm that analyzes and identifies the sections of text that provide the most relevant information to a given query. Microsoft researchers have proposed GraphRAG (Edge et al., 2024), integrating knowledge graphs into RAG to enhance the quality of answers produced by the GenAI component. LoRE: Logit-Ranked Retriever Ensemble for Enhancing Open-Domain Question Answering (Sanniboina et al., 2024) uses an ensemble of diverse retrievers (BM25, FAISS) and offers an answer ranking algorithm that combines the LLM's logits scores, with the retrieval ranks of the passages. Self-Reflective RAG (Akari Asai, 2023), a recent advanced architecture, attempts to improve the retrieval of a simple RAG implementation and thus the generations as a whole by introducing a system that decides when to make a retrieval, otherwise known as adaptive retrieval. Corrective RAG (Shi-Qi Yan, 2024), another recent advanced architecture, is similar to Self-Refective RAG in terms of its dynamic self-analyzing behavior; it is one of the two architectures analyzed in this paper, and therefore described in detail in the following section.

3.1 Corrective RAG

Corrective RAG (Shi-Qi Yan, 2024), or CRAG for short, looks to mitigate hallucinations of low-quality retrieved-context by introducing a lightweight retrieval evaluator, giving a confidence score to work with for the rest of the generation process, which allows the model to adapt its behavior for generations in relation to the confidence score of the retrieved contexts. Similar to Self-Reflective RAG, this confidence score can influence different decisions in the subsequent process by allowing the model to take action against low-quality retrievals, such as augmenting retrieval with a web search or introducing a "decompose-then-recompose" algorithm which, when applied to retrieved documents, allows the system to focus on relevant information and disregard irrelevant information. Figure 2 provides a high-level view of the flow of execution in Corrective RAG: 1) The user submits the query; 2) The retriever component selects documents from the vector database based on the user query; 3) The retrieved documents are graded by the system to determine their relevance relative to the user query; 4) If the documents are relevant, the GenAI (LLM) component generates the answer; 5) If the documents are not relevant, the system attempts to re-write / transform the query, then performs a web search to fulfill the query and submits it to the GenAI component to generate the answer; (6) If there is ambiguity (the score of the retrieved documents coming out of the grading process is neither high nor low, meaning that there is insufficient information to fulfill the query), the system combines the retrieved documents with additional context extracted from the web search and then submits it to the GenAI component to generate the answer.

There is overhead in Corrective RAG that comes with performing web searches or combining documents with web search results, which can, in turn, add to the latency of generations. More importantly, Corrective RAG depends on the quality of the confidence score: If the score is inaccurate, it can yield low-quality generations. Along with retrieval quality concerns, there is also the possibility that bias is introduced when generating answers with web source

retrieval since the retrieval is, in this case, using external, unverified sources as context.

4 EXPERIMENTAL SETUP

4.1 Data Set and Methods

The dataset was initially collected by scraping web pages from the institution's website and annotating questions and their respective contexts. A total of 667 questions and their respective contexts were collected. An excerpt of the dataset can be found in Table 1.

Although RAGAS allows testers to measure RAG performance without relying on ground truth human annotations, in this project's context, we used an evaluation dataset. This was done because no evaluation dataset has been made for the corpus of information we considered (data from our institution).

Table 1: Dataset - Questions and Contexts.

Question	Context
Does Marist offer virtual tours?	Marist offers virtual guided and self-guided campus tours.
What is WCF?	Marist Singers performed at World Choral Fest in Vienna and Salzburg.

The context of all 667 records was fed to the data store of each of the two architectures under consideration (Simple RAG and Corrective RAG)

We experimented on each of the two architectures. For each architecture, we extracted 50 randomly chosen records to perform the experiment runs. This amounted to a total of 50 runs repeated over two architectures for a total of 100 experiments (for details, see section 5 and Table 2).

We used the 50 experiments in each architecture to collect the following metrics: Context Recall, Context Precision, Faithfulness, and Semantic Similarity (see section 2.3 for more details). We then proceeded to compute each metric's means and standard deviations over the 50 random records for each architecture to benchmark both RAG architectures. Due to the structure of the corpus of data used in this project, the question in each sampled record has an answer attached to it (its context). The question in each sampled record was therefore fed to the RAG architecture, simulating a user query, and the context attached to the answer was used as ground truth to evaluate the RAG architecture performance.

For better assessing the experimental results, we added a metric obtained through visual inspection, that we labeled *Face validity*. This was done to get the end-to-end quality of the generation based on the answer. Faithfulness is an adequate holistic metric, but it gauges the quality of the answer only with regard to the retrieved context. It is evidently laborious to compute the metric through visual inspection and can only be done at a small scale, but it supplements faithfulness and provides in this study a relevant measure of the quality of the generations by the RAG architecture under analysis.

The metric, which is calculated using precision, recall, and the F1 measure over the sample of 50 records, was computed for each RAG architecture with these considerations:

- After processing all 50 records with the RAG architecture, each record is inspected, and the question is compared to the generation.
- If the generation correctly answers the question, extracting facts aligned with the ground truth in meaning or factual content, even if phrased differently or not expressing the totality of facts contained in the ground truth, the answer is considered a true positive.
- If the generated answer includes incorrect or irrelevant claims that contradict the ground truth (e.g., hallucinates), the generated answer is considered a false positive.
- If the generation does not entirely answer the question, missing facts contained in the ground truth, then the generation is considered a false negative.
- The count of true positives, false positives, and false negatives is aggregated over all 50 samples in variables TP, FP, and FN.
- Precision, Recall, and the F1 score are calculated with these aggregated counts using the formulas below. F1 is the harmonic mean of precision and recall and, therefore, a more comprehensive, holistic measure.

$$\text{Precision} = \frac{TP}{(TP+FP)} \tag{5}$$

$$\text{Recall} = \frac{TP}{(TP+FN)} \tag{6}$$

$$\text{F1 score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{(\text{Precision} + \text{Recall})} \tag{7}$$

4.2 Computational Platform

The experiments were run on-site using a workstation with the following characteristics: 8-core, 3.2 GHz, 16 GB RAM, 1 TB HD. The software stack was made up of the following components:

Table 2: Development Environment and Tools.

Component	Details
Prog. Platform	Python 3.12
IDE	Visual Studio Code 1.96.2 - Jupyter Notebook Extension 2024.11.0
Vector Storage	ChromaDB 0.6.3, via LangChain (ChromaDB, 2024)
LLM App. Dev.	LangChain 0.3.14 (LangChain, 2024)
LLM	OpenAI's GPT4o-mini (OpenAI, 2024), via LangChain, cloud-based
Vector Embeddings	OpenAI, via LangChain, cloud-based
RAG Metrics	RAGAS 0.2.11 (RAGAS, 2024)

LangChain was selected as the platform of choice to develop and integrate the two RAG architectures under consideration, given its maturity and simplicity (LangChain makes it very easy to set up and access a vector database and retrieval system, and perform LLM API calls with a few lines of code). Simple RAG and the more elaborate Corrective RAG's logic flow were coded using a combination of Python programming and LangChain API calls.

The notebooks with the source code for each of the two RAG architectures under consideration are available from the authors upon request.

5 RESULTS & DISCUSSION

Table 2 displays the results of the assessment of each RAG architecture. As previously mentioned, each experiment was conducted 50 times over the two RAG architectures, each time recording the four RAGAS-based metrics (context recall, context precision, faithfulness, and semantic similarity). The mean metric of each architecture was computed by averaging the 50 outcomes of each metric, together with its standard deviation. Face validity was then assessed by visually inspecting each of the 50 queries and contrasting the generation with the ground truth. This was followed by computing precision, recall, and the F1 score out of the 50 sampled records.

Execution times for each of the metrics over all 50 samples were in the same order of magnitude, with Corrective RAG taking the longest in the case of faithfulness.

In the case of context recall, Simple RAG achieves the highest mean value (0.528) but with the highest variability (stdev=0.365) when compared to Corrective RAG (mean=0.410, stdev=0.325).

In the case of context precision, Corrective RAG (mean=0.956, stdev=0.177) outperforms Simple RAG (mean=0.913, stdev=0.203), having the highest mean and lowest variability for the metric.

Table 3: Results for each metric, $N = 50$.

Metric	Simple RAG	Corrective RAG
Context Recall		
Mean	0.528	0.410
Stdev	0.365	0.325
Time	8 min	8 min
Context Precision		
Mean	0.913	0.956
Stdev	0.203	0.177
Time	5 min	2 min
Faithfulness		
Mean	0.819	0.824
Stdev	0.321	0.213
Time	6 min	14 min
Semantic Similarity		
Mean	0.847	0.881
Stdev	0.072	0.065
Time	25 sec	27 sec
Face Validity		
Precision	0.795	0.666
Recall	0.854	0.941
F1	0.826	0.781

Simple RAG and Corrective RAG had similar faithfulness values (mean=0.819, sd=0.321; mean=0.824, sd=0.213, respectively), with Corrective RAG performing slightly better than Simple RAG.

For semantic similarity, measuring how similar the generation is to the context in terms of meaning, Corrective RAG (mean=0.881, stdev=0.065) outperforms Simple RAG (mean=0.847, stdev=0.072)

In the case of Face Validity, the analysis is more nuanced. Using F1, which combines precision and recall, Simple RAG (F1=0.826) outperforms Corrective RAG (F1=0.781). Still, if we consider precision and recall individually, Corrective RAG has a very high recall (0.941) with the lowest precision value (0.666).

The different RAG processes utilize a context corpus of web-scraped data from various internal and external pages of the College's website. Although each question in the dataset has a correct answer, this answer may be obscured by irrelevant information on the same scraped page. For example, a question about a specific faculty member within a particular department might include context from a general department page listing all faculty members instead of focusing on the individual in question. This can significantly impact evaluation results when using RAGAS metrics, due to how these metrics handle and are influenced by the composition of context documents.

In evaluating context recall, RAGAS calculates it by dividing the ground truth claims within the contexts by the total number of ground truth claims. The ground truth provided to RAGAS is often extensive and contains diverse information, some of which may be irrelevant to the specific question. This complexity negatively impacts the performance of the context recall metric across all architectures. Simple RAG likely performs better than Corrective RAG as it can

pass multiple context documents. Specifically, our implementation retrieves three unique context documents from the vector store, increasing the likelihood of including more ground truth claims. Additionally, Simple RAG - unlike Corrective RAG - lacks conditional logic that can potentially exclude relevant context.

Corrective RAG ensures that context is always used by supplementing or replacing retrieved documents with web searches when necessary. However, the additional context from web searches can introduce irrelevant information, confuse the LLM during the rating process, and further diminish the context recall metric. Overall, Corrective RAG is more susceptible to the composition of the context corpus, which affects its ability to accurately recall relevant context, compared to Simple RAG.

By the same token, Corrective RAG achieves higher context precision than Simple RAG, most likely because it can supplement retrieved-context with web searches. This supplemental information tends to be semantically relevant, especially when it augments existing context documents rather than replacing them. Consequently, Corrective RAG includes more relevant chunks, enhancing context precision, which is an indication of better refinement in its retrieval process.

The faithfulness metric is computed by calculating the number of claims in the generation that are supported by the context divided by the total number of claims in the generation. Corrective RAG outperforms Simple RAG by a slim margin, likely due to its ability to incorporate information from the web. Although web searches can sometimes negatively impact generations for complex or institution-specific questions, the faithfulness metric indicates that web supplementation can be beneficial within this domain. The dataset includes various questions, from specific inquiries about instructors or classes to more general topics like comparing graduate studies in the UK versus the US. When Corrective RAG encounters a general question without a sufficiently relevant context document, a web search effectively supplements or substitutes the missing information, leading to more accurate generations. This capability allows Corrective RAG to be competitive with Simple RAG by grounding generations in relevant contexts. All in all, both architectures have competitive values. A score above 0.8 in both cases indicates that more than 80% of the time, generations are accurate and consistent with the retrieved context, which is generally acceptable for many applications, especially in non-critical domains.

Semantic similarity is calculated using cosine similarity, hence the high values for each architecture. Corrective RAG outperforms Simple RAG, again, due to the ability of Corrective RAG to search the Web and introduce semantically similar context documents, which can positively affect the metric.

Face validity was manually assessed by comparing each question, its ground truth/context, and the generated response to identify true positives, false positives, and false negatives. Simple RAG achieved the highest performance in this metric, likely due to the composition of the context corpus and the absence of conditional logic that otherwise hinders more complex architectures. Simple RAG generates responses directly from the retrieved context without prior evaluation. In contrast, Corrective RAG incorporates information from web searches, which can affect the quality of generations depending on the question. As reported in previous paragraphs, Corrective RAG has a high recall but a comparatively low precision value. Considering that face validity measures the end-to-end quality and completeness of the generation with respect to the query, this means that, on average, Corrective RAG generations provide better alignment with the ground truth in meaning or factual content at the expense of more hallucination. The latter may be due to its ability to search the web to supplement the context and enhance generations, a double-edged sword.

6 CONCLUSION

There is considerable potential in implementing RAG architectures in a higher education setting, but the quality and credibility of responses generated by RAG systems remain substantive issues. Our experimental results show that automated metrics are useful tools for performance evaluation but extensive testing through human intervention is a required step in successful RAG implementation to measure the quality and completeness of the generated answers produced by RAG systems. Furthermore, data quality and the data composition of the corpus of text used for the retriever system are paramount for quality generations. This paper focused on building and testing two variations of RAG architectures using standard open-source tools and a widely recognized commercial LLM. Future work should benchmark other RAG architectures and the use of other large language models, open-source and commercial. The emphasis should be placed on reducing hallucination through improved retrieval strategies, improved quality of the corpus data, and as much as possible, better model

alignment. We recognize that the methodology presented in this paper is general and can be applied to different sectors. However, it offers a valuable guideline for researchers and practitioners to implement and evaluate RAG-based question-answering systems in an educational setting.

REFERENCES

Akari Asai, Zeqiu Wu, e. a. (2023). Self-rag: Learning to retrieve, generate, and critique through self-reflection. https://arxiv.org/abs/2310.11511.

Barnett, S., Kurniawan, S., Thudumu, S., Brannelly, Z., and Abdelrazek, M. (2024). Seven failure points when engineering a retrieval augmented generation system.

Browning, J. G. (2024). Robot lawyers don't have disciplinary hearings—real lawyers do: The ethical risks and responses in using generative artificial intelligence. *Georgia State University Law Review*, 40(4):917–958.

Chen, D., Fisch, A., Weston, J., and Bordes, A. (2017). Reading wikipedia to answer open-domain questions.

ChromaDB (2024). Chromadb: Open-source vector database for ai applications, version 0.6.3. Accessed: 2024-01-07.

Crompton, H. and Burke, D. (2023). Artificial intelligence in higher education: the state of the field. *International Journal of Educational Technology in Higher Education*, 20(1):22.

D-Star-AI (2025). dsrag: An implementation of retrieval-augmented generation (rag). Accessed: 2025-01-09.

Dakshit, S. (2024). Faculty perspectives on the potential of rag in computer science higher education.

Damiano, A. D., Lauría, E. J., Sarmiento, C., and Zhao, N. (2024). Early perceptions of teaching and learning using generative ai in higher education. *Journal of Educational Technology Systems*, 52(3):346–375.

Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S., and Larson, J. (2024). From local to global: A graph rag approach to query-focused summarization.

Es, S., James, J., Espinosa-Anke, L., and Schockaert, S. (2023). Ragas: Automated evaluation of retrieval augmented generation.

Feldman, P., Foulds, J. R., and Pan, S. (2024). Ragged edges: The double-edged sword of retrieval-augmented chatbots.

Gonzalez-Bonorino, A., Lauría, E. J. M., and Presutti, E. (2022). Implementing open-domain question-answering in a college setting: An end-to-end methodology and a preliminary exploration. In *Proceedings of the 14th International Conference on Computer Supported Education - Volume 2: CSEDU*, pages 66–75. INSTICC, SciTePress.

Guu, K., Lee, K., Tung, Z., Pasupat, P., and Chang, M.-W. (2020). Realm: Retrieval-augmented language model pre-training.

Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and tau Yih, W. (2020). Dense passage retrieval for open-domain question answering.

LangChain (2024). Langchain: Building applications with llms through composability. Accessed: 2024-01-07.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

Lyu, W., Wang, Y., Chung, T. R., Sun, Y., and Zhang, Y. (2024). Evaluating the effectiveness of llms in introductory computer science education: A semester-long field study. In *Proceedings of the Eleventh ACM Conference on Learning @ Scale*, L@S '24, page 63–74. ACM.

Modran, H., Bogdan, I. C., Ursuțiu, D., Samoila, C., and Modran, P. L. (2024). Llm intelligent agent tutoring in higher education courses using a rag approach. *Preprints*.

OpenAI (2024). Openai: Advancing ai for everyone. Accessed: 2024-01-07.

RAGAS (2024). Ragas: Robust evaluation for retrieval-augmented generation systems. Accessed: 2024-01-07.

Sanniboina, S., Trivedi, S., and Vijayaraghavan, S. (2024). Lore: Logit-ranked retriever ensemble for enhancing open-domain question answering.

Shi-Qi Yan, Jia-Chen Gu, e. (2024). Corrective retrieval augmented generation. https://arxiv.org/abs/2401.15884.

Thway, M., Recatala-Gomez, J., Lim, F. S., Hippalgaonkar, K., and Ng, L. W. T. (2024). Battling botpoop using genai for higher education: A study of a retrieval augmented generation chatbots impact on learning.

Thüs, D., Malone, S., and Brünken, R. (2024). Exploring generative ai in higher education: a rag system to enhance student engagement with scientific literature. *Frontiers in Psychology*, 15:1474892.

Wong, L. (2024). Gaita: A rag system for personalized computer science education.

Wu, K., Wu, E., Cassasola, A., Zhang, A., Wei, K., Nguyen, T., Riantawan, S., Riantawan, P. S., Ho, D. E., and Zou, J. (2024). How well do llms cite relevant medical references? an evaluation framework and analyses.

Yu, H., Gan, A., Zhang, K., Tong, S., Liu, Q., and Liu, Z. (2024). Evaluation of retrieval-augmented generation: A survey. https://arxiv.org/abs/2405.07437.

Yunfan Gao, Yun Xiong, e. a. (2024). Retrieval-augmented generation for large language models: A survey. https://arxiv.org/abs/2312.10997.