IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 28 January 2026, accepted 8 February 2026, date of publication 16 February 2026, date of current version 19 February 2026.

Digital Object Identifier 10.1109/ACCESS.2026.3664852

RESEARCH ARTICLE

An Empirical Evaluation of Retrieval, Reranking, and Similarity for a Q&A-Based Retrieval Augmented Generation System

HARUN ELKIRAN[1], (Graduate Student Member, IEEE), AND JAWAD RASHEED[2,3,4], (Member, IEEE)

[1]Institute of Graduate Education, Istanbul Sabahattin Zaim University, 34303 Istanbul, Türkiye
[2]Department of Software Engineering, Istanbul Sabahattin Zaim University, 34303 Istanbul, Türkiye
[3]Research Institute, Istanbul Medipol University, 34810 Istanbul, Türkiye
[4]Department of Software Engineering, Istanbul Nisantasi University, 34398 Istanbul, Türkiye

Corresponding author: Jawad Rasheed (jawad.rasheed@izu.edu.tr)

ABSTRACT Retrieval-Augmented Generation (RAG) has emerged as a fundamental paradigm for improving Large Language Models (LLMs) by incorporating external knowledge retrieval. RAG primarily aims to address the hallucination problem in LLMs that rely on extensive knowledge bases. A RAG system depends critically on design choices, including indexing strategies, retrieval methods, similarity metrics, and reranking models. The selection of configuration makes a RAG effective. Although the RAG system has received sufficient attention, there is very limited work on understanding the relative contributions of these components, and their statistical significance remains insufficiently understood. In this study, we conduct a comprehensive empirical evaluation of a modular RAG pipeline by systematically varying index structures, retrievers, rerankers, and similarity metrics. We evaluated performance using standard retrieval metrics such as Recall, Mean Reciprocal Rank, Normalized Discounted Cumulative Gain, and Coverage; generation-oriented quality metrics such as Correctness, Faithfulness, and Relevance; latency; and cost. Statistical robustness is ensured through ANOVA, effect size estimation, and multivariate regression analysis. Based on our results, the retriever and similarity metric choices dominate system performance, yielding statistically significant improvements with p-values less than 10^{-9} for retriever effects on R@1 and Coverage. At the same time, index selection exhibits a negligible impact across most metrics. Reranking primarily affects reranked metrics and downstream correctness, with MiniLM consistently outperforming BGE.

INDEX TERMS Retrieval-augmented generation, information retrieval, retrieval, reranking, similarity/distance metrics.

I. INTRODUCTION

With the introduction of information retrieval (IR) [1], [2] and pre-trained language models [3], interest in developing automated question-answering systems has grown dramatically. Large Language Models (LLMs) [4] have achieved unprecedented success across a wide range of activities, including text generation and complex question answering [5], as well as information retrieval (IR) tasks [6]. However, LLMs have limitations when dealing with extended contexts [7], leading them to rely more heavily on pre-trained information. This shortcoming not only limits their ability to handle longer discourse, such as books or long talks, efficiently, but also increases the likelihood of hallucinations, which occur when the model generates factually incorrect or nonsensical information [8]. Although large-scale pre-trained language models can store massive amounts of knowledge, they struggle to access up-to-date information and provide credible sources. To address this limitation and hallucination problem, approaches that combine information

The associate editor coordinating the review of this manuscript and approving it for publication was Yizhang Jiang.

retrieval with generative models have emerged. Retrieval-Augmented Generation (RAG) uses a neural retriever and a sequence-to-sequence generator to generate outputs from external texts [9].

RAG builds on previous research that employed retrieval to improve question responding and language modeling. RAG provides free-form replies while utilizing non-parametric memory, resulting in better factual accuracy and source citation. This skill is crucial for reducing hallucinations and updating information without retraining the model [10]. RAG takes a more intelligent, evidence-based approach to language generation, greatly reducing the risk of hallucinations and improving the overall quality of generated text [11].

RAG pipelines have been further reinforced by recent developments in dense retrieval using vector embeddings. Even in situations where lexical overlap is minimal, transformer-based [12] embedding models enable efficient similarity-based retrieval by projecting queries and documents into a shared semantic space. Scalable, low-latency retrieval over large corpora is possible with Approximate Nearest Neighbor (ANN) indexing techniques such as HNSW and IVF [13]. Reranking models are frequently used to reorder candidate documents based on deeper semantic interactions between queries and passages, thereby further improving retrieval quality [14], [15]. This greatly increases the correctness and relevance of answers.

Furthermore, it has been demonstrated that query reformulation techniques like Hypothetical Document Embeddings (HyDE) and Hypothetical Passage Embeddings (HyPE) improve retrieval performance, especially in zero-shot or underspecified query settings [16]. These techniques close the semantic gap between user queries and stored documents by generating synthetic representations that capture a query's latent intent. These methods significantly increase downstream answer quality and retrieval accuracy when paired with efficient reranking and generation models.

In this work, we developed and assessed a thorough (RAG) pipeline that systematically examines how indexing strategies, retrieval techniques, reranking models, and similarity metrics affect the overall system performance. The framework incorporates neural reranking using pre-trained cross-encoder models, multiple indexing (HNSW, IVF, and ScaNN), and a variety of retrieval strategies (Fusion, Hierarchical, HyDE, and HyPE) with dense vector embeddings. Standard IR metrics are used to assess performance throughout the retrieval, reranking, and generation stages. We perform comprehensive statistical testing, including analysis of variance (ANOVA), effect size estimation, and regression-based modeling, to assess the statistical significance and practical impact of each system component and ensure robustness and reliability. The key contributions of this study are as follows:

1) Comprehensive empirical evaluation of multiple vector indexing methods, retrieval strategies, reranking models, and similarity metrics within a unified RAG framework.
2) Statistically grounded analysis using ANOVA, effect size measures, and regression modeling to identify significant performance drivers across retrieval, reranking, and generation stages.
3) Practical insights for dense retrieval-based RAG systems under controlled general-domain settings, highlighting trade-offs between accuracy, latency, and cost, and providing evidence-based guidance for selecting indexing, retrieval, and reranking configurations.

The remainder of this paper is organized as follows. Section II reviews related work in information retrieval, retrieval-augmented generation, and vector databases. Section III describes the system architecture, embedding generation, indexing strategies, retrieval and reranking methods, and text generation process. Section V reports quantitative results and statistical analyses, including significance testing and effect size estimation. Finally, Section VII discusses key findings, limitations, and directions for future work.

II. RELATED WORK

Retrieval Augmented Generation (RAG) has emerged as a foundational paradigm for addressing the limitations of large language models in knowledge-intensive tasks [17]. While early RAG systems demonstrated clear gains in factual grounding, subsequent research has shown that retrieval quality, reranking strategies, indexing mechanisms, and query adaptation play a decisive role in overall system performance. Recent research has shown that RAG effectiveness can be significantly increased by improving the retrieval and reranking stages. Study [18] discusses enhancements in the RAG pipeline through advanced optimization techniques. Study [18] identifies limitations in the basic RAG system, including issues with chunking, hallucinations, and reliance on augmented content for knowledge-intensive tasks. Also in [18], advanced strategies for chunking, vectorization, and search processes are proposed to improve retrieval and generation tasks. This study [18] also shows that the techniques such as reranking, filtering, query transformation, query routing, and response synthesis can enhance the relevance, coherence, and accuracy of generated responses. All the elements of a RAG pipeline (indexing, retrieval, and reranking) play an important role in RAG performance, as survey [19] compiles and analyzes benchmark results across standard datasets such as Natural Questions, TriviaQA, ELI5, FEVER, and HotpotQA. It evaluates various architectures, including DPR, FiD, REALM, RAG-Sequence, Atlas, and GTR. A comparative evaluation highlights trade-offs between retrieval cost, generation accuracy, document redundancy, and interpretability across different settings. The paper [19] identifies key open challenges in RAG, including the need for unified end-to-end optimization and robust strategies to mitigate hallucination. Authors in [20] perform the comparative analysis of state-of-the-art indexing methods on

three multihop datasets (MuSiQue, 2WikiMultiHopQA, and HotpotQA) and based on the results, their proposed SiReRAG consistently outperforms with an average improvement of 1.9% in F1 scores. SiReRAG achieved approximately 5% higher average F1 scores than RAPTOR. SiReRAG also showed an improvement of up to 8.3% in F1 scores on MuSiQue compared to HippoRAG. Apart from indexing, another important aspect of a RAG pipeline is the rerankers. The reranking framework proposed in [21] improves retrieval quality in RAG systems. It effectively integrates user intent prediction to optimize the RAG process. Evaluation on two benchmark datasets demonstrates enhanced personalization and relevance in dynamic user query scenarios. A lot of the technologies involved in the RAG system depend on the language. Researchers in [22] conducted a comparative analysis of rerankers with first-stage retrieval methods in Polish. Based on the results, most rerankers failed to outperform the first-stage retrieval methods, with only two models achieving higher average NDCG10 values. The best-performing model, mt5-13b-mmarco-100k, set a new state-of-the-art for reranking in Polish, outperforming existing models with up to 30 times more parameters. It was also observed in [22] that some datasets, such as PolEval and MAUPQA, were easier for rerankers, whereas others, such as Web Datasets and BEIR-PL, posed significant challenges due to their data diversity. [15] shows that reranking is crucial for RAG with Large Language Models, as G-RAG outperforms PaLM 2 significantly. Also, not just reranking, but other parameters such as chunking and indexing are also very important for RAG performance, as shown in [23]. Response fidelity with dynamic chunking and re-ranking improved RAG performance, also showing the significant impact of index choice on performance metrics. In RAG systems, there are always challenges in text retrieval pipelines, particularly trade-offs among model size, ranking accuracy, and system requirements such as indexing and serving latency/throughput. Based on the results from the [24], smaller embedding models like snowflake-arctic-embed-l and NV-EmbedQA-e5-v5 showed considerable improvement in ranking accuracy when combined with cross-encoders. The NV-RerankQA-Mistral-4B-v3 reranker achieved a significant accuracy increase of approximately 14% compared to other rerankers in the benchmark. For the larger NV-EmbedQA-Mistral7B-v2 embedding model, only the NV-RerankQA-Mistral-4B-v3 reranker improved its accuracy. The NV-RerankQA-Mistral-4B-v3 provided the highest ranking accuracy across all datasets evaluated. The paper [25] aims to provide insights for researchers and practitioners on leveraging RAG effectively to improve text generation tasks. [25] reviews the foundational principles and architecture of retrieval augmented generation (RAG) in natural language processing. [25] highlights the two-step RAG process, which involves retrieving relevant information from the input query and generating text informed by both the query and the retrieved knowledge. It emphasizes integrating retrieved knowledge into the generative model to enhance text quality and contextual relevance.

The studies discussed in this section show that although significant advancements have been made in improving retrieval, reranking, and indexing strategies for RAG systems, no single strategy consistently prevails across datasets, languages, and system limitations. Improvements in relevance or accuracy may entail trade-offs in latency, cost, or scalability, and performance gains are often dataset-dependent. Prior studies are more focused on benchmark-driven evaluations [26], [27], which emphasize absolute performance across datasets or architectures; this study adopts a system-level empirical perspective. By applying factorial ANOVA, effect size analysis, and regression modeling, we provide statistically grounded insights into which RAG components dominate system behavior. This allows us to move beyond descriptive comparisons toward explanatory conclusions that inform practical system design.

III. METHODOLOGY

This study assesses how chunking, indexing, retrieval, and reranking strategies affect question-answering performance in a structured RAG pipeline. Using several approximate nearest neighbor techniques, documents are first chunked and converted into dense embeddings, which are then stored and indexed. Using various similarity metrics and retrieval techniques, queries are embedded and compared to stored representations. Before being sent to a lightweight generative model for answer synthesis, retrieved candidates undergo cross-encoder reranking. Ranking, quality, efficiency, and statistical significance analyses are used to assess system performance and ensure both efficacy and robustness across configurations. Complete methodological flow of the study is shown in Figure 1.

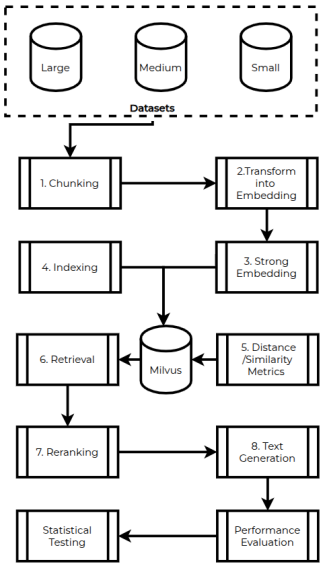

FIGURE 1. Methodology flowchart diagram.

A. DATASETS

The first phase in this study is data collection. This study utilized the Stanford Question Answering Dataset (SQuAD) [28]. It is commonly used as a standard for question-answering tasks in machine learning. SQuAD was developed by human annotators from Wikipedia articles. Each dataset instance consists of a Wikipedia passage, a corresponding question, and one or more ground-truth answers. SquAD consists of 100,000 instances. The small, medium, and large datasets [29] are constructed by random sampling 10k, 30k, and 100k instances, respectively, while ensuring proportional coverage of topics and question types. This scaling strategy enables controlled analysis of dataset size effects without introducing distributional bias**.**

B. CHUNKING

RAG systems rely heavily on document chunking, the technique of splitting texts into coherent chunks. More formally, Chunking is a transformation $T : D \rightarrow P$ that maps a document D to a set of passages $P = \{p_1, p_2, \ldots\}$. The transformation is designed such that the semantic information contained in D is preserved within the passage set P. The transformation may be reversible, meaning that the original document can potentially be reconstructed from P, although reversibility is not a strict requirement [30]. Recent studies suggest that chunking methods affect downstream task performance [31], and that LLMs are sensitive to passage style and organization [32]. This study used chunks with a fixed window size of 1,000 and a character overlap of 200. This selected ratio provides a balanced granularity with contextual completeness. This study's overlap set preserves information around chunk borders. Once chunking is performed, it yields a representation that enables efficient embeddings for downstream tasks.

C. TRANSFORM INTO EMBEDDINGS

Numerical representations of text, pictures, or other data types that capture semantic value in a high-dimensional vector space are called embeddings. Embeddings are a very important part of an RAG system; they provide appropriate representations of chunks for storage, retrieval, and downstream tasks. In this study, each passage p_i from the chunking phase is then processed by OpenAI's `text-embedding-large-3`[1] model. The model first tokenizes the text into a sequence of subword tokens $(t_1, t_2, \ldots, t_m)$ using a byte-pair encoding (BPE) tokenizer. A deep transformer encoder maps these tokens to contextual representations, creating a series of hidden states that capture the passage's syntactic and semantic dependencies. A pooling operation over the final-layer token representations yields a fixed-dimensional embedding vector $\mathbf{e}_i \in \mathbb{R}^{3072}$. To facilitate effective semantic retrieval and comparison across chunked document representations, the resulting embedding space is optimized so that semantically similar passages are closer to each other in the vector space.

[1] https://platform.openai.com/docs/models/text-embedding-3-large

D. STORING EMBEDDINGS

The dense numerical representations (vector embeddings) obtained in the previous step enable similarity-based retrieval by encoding the semantic meaning of unstructured data. The embeddings produced in this work are stored in Milvus [33], a vector database designed for extensive semantic search. Fast and precise similarity queries are made possible by Milvus's scalable high-dimensional vector storage and effective Approximate Nearest Neighbor (ANN) indexing. Real-time embedding insertion, retrieval, and filtering are enabled by its distributed, cloud-native architecture, making it ideal for the RAG system.

E. INDEXING

Efficient similarity search over high-dimensional embedding spaces requires specialized indexing strategies to balance search accuracy, latency, and memory consumption. In this system, three approximate nearest neighbor (ANN) [34] indexing approaches are considered: HNSW, IVF, and ScaNN.

1) HIERARCHICAL NAVIGABLE SMALL WORLD (HNSW)

HNSW arranges embeddings into a multi-layer graph structure, with edges connecting semantically close neighbors and each node representing a vector. Logarithmic-time complexity and high recall are made possible by the search's traversal of the graph from higher to lower layers during query time. Despite having a higher memory overhead, HNSW is especially useful for low-latency applications.

2) INVERTED FILE INDEX (IVF)

IVF uses coarse quantization to divide the embedding space into a predetermined number of clusters. The number of distance calculations is greatly reduced by assigning each vector to its closest cluster centroid and limiting similarity searches to a subset of relevant clusters. With a manageable trade-off between speed and accuracy, IVF is ideal for large-scale datasets where scalability and memory efficiency are crucial.

3) SCALABLE NEAREST NEIGHBORS (ScaNN)

Space partitioning, vector quantization, and optimized distance scoring are all combined in ScaNN's hybrid methodology. ScaNN is appropriate for high-throughput, large-scale semantic search scenarios because it balances retrieval accuracy and computational efficiency by selectively searching promising partitions and re-ranking compressed representations.

F. DISTANCE / SIMILARITY METRICS

Similarity and distance metrics are used in RAG to estimate how relevant a chunk of text is to a user's query [35], [36]

[37]. These ratings help the retrieval system determine which passages to feed into a language model for answer generation. The choice of metric is greatly influenced by the embedding model's training method. Embedding models are tailored to arrange their output vectors in ways that are consistent with specified distance measures. It is essential to index and store vectors. In this study, we have used three metrics.

1) COSINE SIMILARITY

The angular similarity between two embedding vectors $\mathbf{x}$, $\mathbf{y} \in \mathbb{R}^d$ is measured using cosine similarity, regardless of their magnitudes. It is defined by the equation 1.

$$\text{sim}_{\cos}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\|_2 \, \|\mathbf{y}\|_2}, \tag{1}$$

In equation 1 $\mathbf{x} \cdot \mathbf{y}$ shows the dot product and $\|\cdot\|_2$ also denotes the Euclidean norm.

2) EUCLIDEAN DISTANCE

Equation 2 defines Euclidean distance, which measures the straight-line distance between two vectors in the embedding space. This metric is frequently used when the absolute distance in the embedding space is significant, as it is sensitive to vector magnitude.

$$d_{\text{Euc}}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2 = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}. \tag{2}$$

3) INNER PRODUCT

Equation 3 calculates inner product similarity, which evaluates the alignment of two vectors. Unlike cosine similarity, the inner product does not normalize vector lengths. It thereby captures both directional similarity and magnitude, making it appropriate for applications such as maximum relevance score in retrieval systems.

$$\text{sim}_{\text{IP}}(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^{d} x_i y_i. \tag{3}$$

G. RETRIEVAL

Retrieval is the Information Retrieval task that identifies the most relevant documents or passages from the embedding vector space in response to a query. Given a query q, it is first transformed into a vector representation $\mathbf{q}$ using the same embedding model as the indexed data based on the indexing techniques used in this study. Similarity between the query embedding and stored embeddings is then computed using a predefined distance/similarity metric, based on this, the top-k most relevant texts are retrieved. Advanced retrieval strategies further enhance recall and precision of a RAG system. In this study, we have used four retrieval strategies: fusion, hierarchical, HyPe, and HyDe.

1) FUSION

Fusion-based retrieval combines results from different retrieval strategies or representations to improve overall recall and robustness. Typically, scores from dense vector retrieval and sparse keyword-based retrieval are added together using weighted summation or rank-based fusion, as shown in equation 4. In equation 4, $s_i(d)$ denotes the score of document d from the i-th retriever, and α_i represents its corresponding weight. Fusion leverages the strengths of various retrieval methods to produce more reliable results [38].

$$s(d) = \sum_{i=1}^{n} \alpha_i s_i(d), \tag{4}$$

2) HIERARCHICAL

Hierarchical retrieval operates on multi-level document representations, such as document- and chunk-level embeddings. An initial coarse retrieval step finds relevant documents, followed by a detailed search within their associated passages, as shown in equation 5. In equation 5, $\mathbf{q}$ is the query embedding, $\mathbf{e}_D$ denotes document-level embeddings, and $\mathbf{e}_p$ represents passage-level embeddings. This approach boosts efficiency while keeping retrieval precision.

$$D' = \arg\max_{D} \text{sim}(\mathbf{q}, \mathbf{e}_D), \quad P' = \arg\max_{p \in D'} \text{sim}(\mathbf{q}, \mathbf{e}_p), \tag{5}$$

3) HyPE

HyPE (Hypothetical Passage Embeddings) improves retrieval by creating a hypothetical passage that represents the ideal answer to a given query. This generated passage is embedded and used as a retrieval query, as shown in equation 6. In equation 6, $\hat{p}$ is the generated hypothetical passage and $E(\cdot)$ denotes the embedding function. By aligning retrieval with the expected answer structure, HyPE boosts semantic matching and recall [16].

$$\mathbf{q}_{\text{HyPE}} = E(\hat{p}), \tag{6}$$

4) HyDE

HyDE (Hypothetical Document Embeddings) follows a similar approach, generating a hypothetical document rather than a passage. The embedding of this synthetic document is used to retrieve relevant real documents, as shown in equation 7. In equation 7, $\hat{D}$ is a generated hypothetical document. HyDE is particularly effective in zero-shot or low-resource settings because it reduces reliance on exact query wording by capturing hidden intent [39].

$$\mathbf{q}_{\text{HyDE}} = E(\hat{D}), \tag{7}$$

H. RERANKING

A reranker model, also known as a cross-encoder, generates a similarity score for a query-document pair. We use this score to arrange documents or outputs by their relevance to our query. Rerankers are best used as a second-stage ranking retrieval step. The first heavy lifting is done via rapid, efficient embedding-based vector search (retriever), which retrieves a set of documents from a large knowledge base but may lack the high accuracy required in some use cases. Rerankers then refine a list of candidates, offering just the

most relevant documents to increase retrieval accuracy. In this study, we have used two rerankers, BGE and miniLM.

1) BGE

We employed `BAAI/bge-reranker-base`[2] in this study, which supports both Chinese and English. Because BGE is a cross-encoder model, it is less efficient but more accurate. In contrast to the embedding approach, the BGE reranker takes questions and documents as input and outputs similarity scores directly rather than via embeddings. Cross-encoders are frequently used to re-rank top-k documents produced by other basic models to balance accuracy and time cost. For instance, the top 100 relevant documents may be retrieved using the embedding model during the retrieval phase, and the final top-3 results can be obtained by the reranker model using the BGE reranker.

2) miniLM

In this study, for miniLM reranker, we used `cross-encoder/ms-marco-MiniLM-L-6-v2`.[3] This model is trained on the MS MARCO dataset and optimized for passage-level relevance estimation, enabling it to capture contextual alignment and semantic nuances that are not accessible during the initial dense retrieval stage. This re-ranking step significantly improves precision by prioritizing the most relevant passages to the query intent. Following the initial retrieval stage, a re-ranking phase refines the relevance of the top-k retrieved candidates. Given a query q and a candidate passage p_i, the model encodes the concatenated input sequence $[q; p_i]$ and produces a relevance score shown in equation 8

$$s_i = f_{\text{CE}}(q, p_i), \tag{8}$$

In equation 8 $f_{\text{CE}}(\cdot)$ denotes the cross-encoder scoring function. The retrieved candidates are then rearranged according to s_i, yielding a refined ranking shown in equation 9

$$\mathcal{R} = \text{sort}(\{p_i\}_{i=1}^{k}, s_i). \tag{9}$$

I. TEXT GENERATION

In this study, the `gpt-4o-mini` model is used to generate the final response. This model produces a logical, natural-language response to the user query using the top-ranked chunks from the reranking stage as contextual input. The model minimizes hallucinations while producing grounded and context-aware responses by conditioning generation on retrieved external knowledge. Low-latency generation is enabled by `gpt-4o-mini`'s lightweight, effective design, making it suitable for interactive applications that demand both accuracy and responsiveness. The gpt-4o-mini model was selected as the generation component because our objective is not to optimize generation quality, but to isolate and evaluate the relative contributions of retrieval, similarity metrics, indexing, and reranking components. Using a lightweight and cost-efficient generator serves that purpose. If we employ larger or more smart generator (e.g., GPT-4, Claude, or LLaMA-based models), it can improve the absolute correctness and fluency scores, but it would not change the statistical significance or effect sizes observed for retrievers, similarity metrics, or indexing strategies, used in this study prior to generation.

[2]https://huggingface.co/BAAI/bge-reranker-base
[3]https://huggingface.co/cross-encoder/ms-marco-MiniLM-L6-v2

J. PERFORMANCE EVALUATION

System performance is evaluated across the retrieval, reranking, and generation stages. We use a mix of ranking-, coverage-, quality-, and efficiency-related metrics. These metrics assess retrieval accuracy, answer quality, robustness, and system efficiency.

1) RECALL@1 (R@1)

Recall@1 measures whether the most relevant document or passage is ranked first. It reflects the system's ability to produce the correct result immediately. This measure is crucial for reranking and generation performance.

2) MEAN RECIPROCAL RANK (MRR)

MRR evaluates the average inverse rank of the first relevant result across all queries. It shows how early the correct answer appears in the ranked list and is sensitive to improvements beyond the top position.

3) NORMALIZED DISCOUNTED CUMULATIVE GAIN AT 1 (nDCG@1)

nDCG@1 measures ranking quality by considering both relevance and position, normalized against an ideal ranking. By focusing on the top-ranked result, it highlights precision in important retrieval scenarios.

4) COVERAGE

Coverage quantifies the proportion of queries where the system successfully retrieves at least one relevant document. This metric reflects retrieval robustness and the ability to avoid empty or irrelevant result sets.

5) CORRECTNESS

Correctness evaluates whether the generated response includes accurate information based on the retrieved context. It captures the system's reliability and is essential for practical use.

6) FAITHFULNESS

Faithfulness measures how well generated responses stick to the retrieved evidence. High faithfulness means less hallucination and better alignment between the retrieval and generation stages.

7) RELEVANCE

Relevance assesses how effectively the generated output answers the user's query. Unlike correctness, which focuses

on factual accuracy, relevance emphasizes how well the response meets user intent and its usefulness.

8) LATENCY

Latency is the average time from start to finish, including retrieval, reranking, and generation. This metric is key for evaluating system usability in real-time applications.

9) COST

Cost measures the average computational or API-related expense per query. It offers insight into the economic efficiency of different configurations and helps analyze trade-offs between performance and operational cost.

K. STATISTICAL TESTING

To assess whether observed performance differences are statistically significant, we apply factorial statistical analysis across four experimental factors: index type, retrieval strategy, reranker, and similarity metric. To achieve this, we have performed following performed following statistical test.

1) ANALYSIS OF VARIANCE (ANOVA)

One-way and multi-factor ANOVA tests are used to evaluate whether performance variations across factor levels are significant. The ANOVA F-statistic is defined using equation 10. A low p-value ($p < 0.05$) indicates statistically significant differences between groups.

$$F = \frac{\text{Between-group variance}}{\text{Within-group variance}} \tag{10}$$

2) EFFECT SIZE (ETA SQUARED)

To quantify the magnitude of observed effects, eta squared (η^2) is computed using equation 11, where SS denotes the sum of squares. Large η^2 values for retrievers and similarity metrics indicate a strong practical impact on system performance.

$$\eta^2 = \frac{\text{SS}_{\text{factor}}}{\text{SS}_{\text{total}}} \tag{11}$$

3) REGRESSION ANALYSIS

Multivariate linear regression is applied to estimate the directional influence of individual factor levels. Regression coefficients (β) quantify the contribution of each factor from equation 12. Positive β values indicate performance improvements, while negative values reflect degradation.

$$y = \beta_0 + \sum_j \beta_j x_j + \epsilon \tag{12}$$

IV. SYSTEM IMPLEMENTATION

All experiments were implemented in Python using the LangChain framework with Milvus as the vector database backend. Experiments were conducted on a MacBook Pro with a 2.3 GHz 8-core Intel Core i9 CPU, 32 GB of DDR4 (2667 MHz) memory, and macOS 15.6.1 operating system. Dense embeddings are generated using text-embedding-3-large, while answer generation relies on gpt-4o-mini with deterministic decoding. Latency and API costs are recorded separately for retrieval, reranking, and generation using OpenAI callbacks. We have designed the RAG pipeline to be fully modular and configurable via command-line interfaces, enabling controlled variation of different parameters such as indexing strategy (HNSW, IVF, SCANN), similarity metric (cosine, inner product, L2), retriever (HyDE, HyPE, Fusion, Hierarchical), reranker (LLM-based or cross-encoder), and dataset size. The System prompt used in this study is shown in Figure 2.

System Prompt:
You are an assistant for question-answering tasks. Use the retrieved context to answer the question. If you do not know the answer, state that you do not know. Use at most three sentences and keep the response concise.

Context:
`{context}`

User Query:
`Question: {question}`

FIGURE 2. Prompt template used for answer generation in all experiments.

Table 1 shows the Key implementation parameters used in all experiments. This information can lead to successful reproducibility.

TABLE 1. Key implementation parameters used in all experiments.

Component	Setting
Embedding model	text-embedding-3-large
Generation model	gpt-4o-mini (temperature = 0)
Vector database	Milvus
Chunk size / overlap	1000 / 200 characters
Retrieval depth (k)	10
Dataset sizes	10k, 30k, 100k chunks
CPU	2.3 GHz 8-Core Intel Core i9 CPU
RAM	32 GB of DDR4 (2667 MHz)
GPU	None
Prompt	Figure 2

V. RESULTS

A. INDEXING PERFORMANCE ANALYSIS

Table 2 presents the mean performance metrics for the three indexing strategies: HNSW, IVF, and ScaNN. Among the evaluated methods, HNSW achieves the highest retrieval effectiveness, with a Recall@1 of 0.578, Mean Reciprocal Rank (MRR) of 0.654, and nDCG@1 of 0.578. In comparison, IVF shows slightly lower scores for R@1 and MRR at 0.564 and 0.638, respectively, while ScaNN performs marginally better than IVF, with an MRR of 0.640. Coverage during retrieval follows a similar trend, with HNSW achieving the highest coverage of 0.796, compared to 0.775 for IVF and 0.778 for ScaNN. After reranking, the relative ordering of the indexing methods remains consistent, with HNSW maintaining superior performance, achieving an MRR of 0.654, over IVF (0.639) and ScaNN (0.640), indicating stable

retrieval quality across pipeline stages. In terms of generation quality, HNSW again outperforms the alternatives, achieving a correctness score of 0.783, a faithfulness score of 0.930, and a relevance score of 0.866. IVF and ScaNN show slightly reduced correctness with scores of 0.771 and 0.772, respectively, for relevance; both models were at 0.856, while faithfulness remains high and comparable across all methods, i.e., above 0.927. HNSW exhibits the highest mean latency at 3.176 seconds, whereas IVF and ScaNN achieve lower response times of 2.838 and 2.815 seconds, respectively. This latency analysis reveals a trade-off between effectiveness and efficiency. The average cost per query remains constant across all indexing methods at 1.16×10^{-4}, indicating that performance differences are primarily driven by retrieval behavior rather than computational cost.

TABLE 2. Comparison of mean performance metrics across indexing methods.

Mean Indexes Performance Metrics			
Metrics	hnsw	ivf	scann
R@1_retrieval	0.578	0.564	0.566
MRR_retrieval	0.654	0.638	0.640
nDCG@1_retrieval	0.578	0.564	0.566
Coverage_retrieval	0.796	0.775	0.778
R@1_reranked	0.578	0.565	0.566
MRR_reranked	0.654	0.639	0.640
nDCG@1_reranked	0.578	0.565	0.566
Coverage_reranked	0.796	0.775	0.778
Correctness	0.783	0.771	0.772
Faithfulness	0.930	0.927	0.928
Relevance	0.866	0.856	0.856
Latency Mean	3.176	2.838	2.815
Cost Mean	0.000116	0.000116	0.000116

**Note: Costs reflect API usage only*

Table 3 compares the performance of four retrieval strategies across retrieval, reranking, and generation-oriented metrics. Among all methods, HyPE consistently achieves the strongest retrieval effectiveness, obtaining the highest R@1 of 0.686 and MRR of 0.766, outperforming HyDE by 0.087 in R@1 and 0.073 in MRR. HyDE also shows strong performance, particularly in coverage, achieving 0.858 compared to 0.717 for Fusion and 0.652 for Hierarchical retrieval. After reranking, HyDE maintains a high MRR of 0.701, while HyPE remains superior at 0.741. In terms of generation quality, HyPE achieves the highest correctness with a score of 0.846. Hype achieved the faithfulness score of 0.941 and the relevance score of 0.918, indicating improved grounding and answer alignment. Although HyDE incurs higher latency and cost, HyPE provides the best balance between effectiveness and efficiency, with the lowest cost of 0.000097 and a competitive latency of 2.135 seconds.

Table 4 compares the performance of the BGE and MiniLM rerankers across retrieval, reranking, and generation metrics. While both models exhibit identical first-stage retrieval performance with R@1 and MRR scores of 0.570 and 0.644, respectively, substantial differences emerge

TABLE 3. Mean performance metrics across different retrieval strategies.

Mean Retriever Performance Metrics				
Metrics	fusion	hierarchical	hyde	hype
R@1_retrieval	0.491	0.503	0.599	0.686
MRR_retrieval	0.559	0.558	0.693	0.766
nDCG@1_retrieval	0.491	0.503	0.599	0.686
Coverage_retrieval	0.717	0.652	0.858	0.904
R@1_reranked	0.539	0.468	0.619	0.653
MRR_reranked	0.604	0.531	0.701	0.741
nDCG@1_reranked	0.539	0.468	0.619	0.653
Coverage_reranked	0.717	0.652	0.858	0.904
Correctness	0.756	0.687	0.812	0.846
Faithfulness	0.932	0.907	0.934	0.941
Relevance	0.843	0.783	0.893	0.918
Latency Mean	2.586	2.210	4.841	2.135
Cost Mean	0.000098	0.000099	0.000171	0.000097

**Note: Costs reflect API usage only*

after reranking. MiniLM significantly outperforms BGE in reranking with R@1 of 0.687 and MRR of 0.725. This indicates strong relevance modeling at the cross-encoder stage. Improvements are also observed in answer quality metrics, with MiniLM achieving higher correctness, faithfulness, and relevance with scores of 0.803, 0.939, and 0.879, respectively. In addition, MiniLM demonstrates lower average latency compared to BGE, while both models incur identical computational cost.

TABLE 4. Comparison of mean performance metrics for BGE and MiniLM rerankers.

Mean Reranker Performance Metrics		
Metrics	bge	minilm
R@1_retrieval	0.570	0.570
MRR_retrieval	0.644	0.644
nDCG@1_retrieval	0.570	0.570
Coverage_retrieval	0.783	0.783
R@1_reranked	0.452	0.687
MRR_reranked	0.564	0.725
nDCG@1_reranked	0.452	0.687
Coverage_reranked	0.783	0.783
Correctness	0.748	0.803
Faithfulness	0.918	0.939
Relevance	0.839	0.879
Latency Mean	3.125	2.761
Cost Mean	0.000116	0.000116

**Note: Costs reflect API usage only*

Table 5 reports the impact of different distance and similarity metrics on system performance. Inner product similarity achieves the strongest overall retrieval results, with an MRR of 0.7087 and nDCG@1 of 0.6328, outperforming cosine similarity. Cosine similarity remains competitive, delivering slightly lower retrieval effectiveness but comparable generation quality with the faithfulness score of 0.9332. In contrast, L2 distance performs substantially worse across all metrics, with an MRR of only 0.5213 and reduced coverage of 0.6758.

TABLE 5. Comparison of distance and similarity metrics across retrieval, reranking, and generation performance.

Mean Distance/Similarity Metrics Performance Metrics			
Metrics	cosine	inner_product	l2
R@1_retrieval	0.625856	0.632755	0.450231
MRR_retrieval	0.701346	0.70872	0.521317
nDCG@1_retrieval	0.625856	0.632755	0.450231
Coverage_retrieval	0.831597	0.841574	0.675787
R@1_reranked	0.602164	0.612396	0.494167
MRR_reranked	0.682632	0.693064	0.557368
nDCG@1_reranked	0.602164	0.612396	0.494167
Coverage_reranked	0.831597	0.841574	0.675787
Correctness	0.80188	0.809325	0.715372
Faithfulness	0.933191	0.935576	0.916488
Relevance	0.879656	0.885685	0.812483
latency_mean	2.965588	3.026272	2.836867
cost_mean	0.000116	0.000116	0.000117

**Note: Costs reflect API usage only*

Figure 3 analyzes indexing methods HNSW, IVF, and ScaNN. HNSW provides slightly better effectiveness, with higher relevance (0.86) and coverage compared to IVF and ScaNN. However, this comes with a marginally higher latency of 3.18 seconds on average. ScaNN offers the lowest latency of 2.81 seconds while maintaining comparable correctness and faithfulness, making it a strong choice for latency-sensitive applications. Cost and token usage remain stable across all index types, indicating indexing primarily affects latency–accuracy trade-offs.

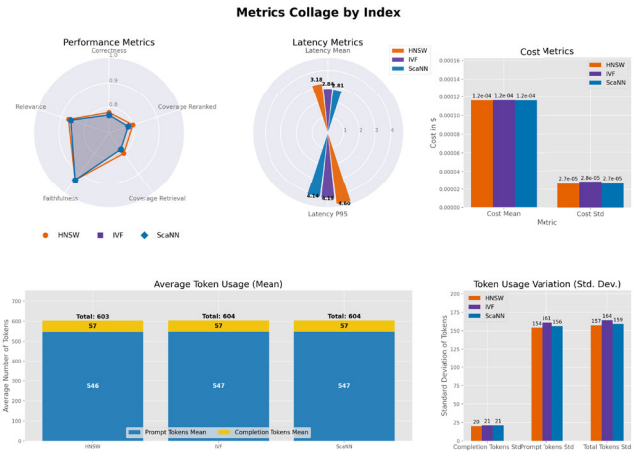

FIGURE 3. Impact of indexing methods on system performance.

Figure 4 compares cosine similarity, inner product, and L2 distance across performance, latency, cost, and token usage. The inner product achieves the strongest overall retrieval quality, with higher correctness and relevance of 0.81 and 0.89, respectively, while L2 consistently underperforms, showing lower correctness and relevance. Latency differences are small, with cosine yielding the lowest mean latency of 2.97 seconds and inner product a slightly higher latency of 3.03 seconds. Cost and token usage remain nearly identical across metrics

Figure 5 compares Fusion, Hierarchical, HyDE, and HyPE retrieval strategies. HyPE consistently delivers the best results, achieving the highest correctness, relevance, and coverage scores of 0.85, 0.92, and 0.90, respectively. HyDE shows competitive effectiveness but incurs significantly higher latency and token usage, indicating a trade-off between recall and efficiency. Fusion and Hierarchical approaches lag behind in both relevance and coverage, underscoring the benefits of hypothetical embedding-based retrieval strategies.

Figure 6 highlights the impact of reranking models. MiniLM clearly outperforms BGE in effectiveness metrics, achieving higher correctness, faithfulness, and relevance. MiniLM also offers a lower mean latency of 2.76 seconds compared to BGE's 3.12 seconds, while the cost remains identical. These results demonstrate that lightweight cross-encoders, such as MiniLM, can deliver both higher accuracy and greater efficiency in second-stage reranking.

Table 6 reports the results of an ANOVA test examining the effects of Index type, Retriever strategy, Reranker, and Similarity Metric across all evaluation metrics. The results indicate that retrieval strategy is the most statistically significant factor affecting retrieval effectiveness. For instance, retrieval-stage performance metrics such as R@1 and MRR exhibit highly significant effects for the retriever factor. In contrast, index type does not exhibit statistically significant effects across most metrics, such as $R@1_{retrieval} p$ value is 0.889, and for $MRR_{retrieval} p$ value is 0.862, indicating that performance differences among HNSW, IVF, and ScaNN are largely negligible when compared to higher-level architectural choices. Similarity metrics show consistent statistical significance across both retrieval and reranking. This highlights their critical role in embedding-space matching. Latency and cost metrics are dominated by the retriever factor, with extremely large F-statistics, confirming that retrieval design choices primarily determine system efficiency.

Table 7 summarizes effect sizes using η^2 to quantify the magnitude of each factor's influence. The retriever has a large effect on Coverage and Correctness ($\eta^2 \approx 0.31$), indicating that the retrieval strategy accounts for nearly one-third of the observed variance in these metrics. Similarity metrics also demonstrate large effects on Coverage ($\eta^2 = 0.176$) and Correctness ($\eta^2 = 0.156$), underscoring their substantive impact beyond statistical significance alone. In contrast, index type consistently shows negligible effect sizes ($\eta^2 < 0.01$), corroborating the ANOVA results and indicating that index choice has minimal practical influence on final system performance.

Table 8 reveals that retrieval strategy and similarity metric are the dominant contributors to retrieval and coverage performance. Among similarity metrics, inner product exhibits the strongest positive effect on R@1 and coverage, where β for R@1 retrieval is 0.030, p value is less than 10^{-5}, while L2 distance consistently degrades performance with large

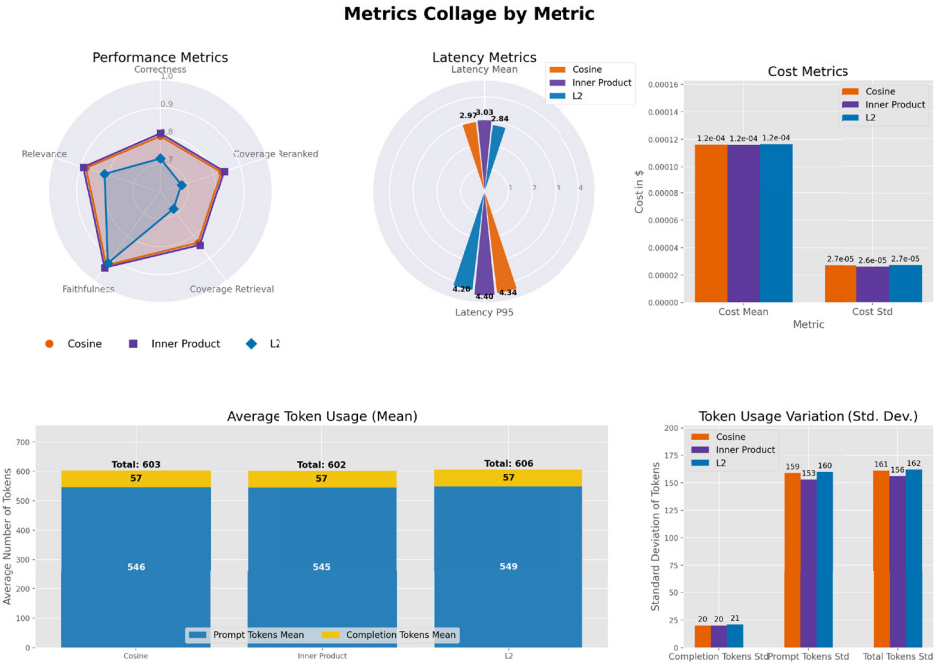

FIGURE 4. Comparison of distance and similarity metrics across performance, latency, cost, and token usage.

TABLE 6. Factorial ANOVA results showing the statistical significance of Index type, Retrieval strategy, Reranker, and Similarity Metric across all evaluation metrics.

Metrics	F_stat	p_value	F_stat	p_value	F_stat	p_value	F_stat	p_value
	Index		Retriever		Reranker		Metric	
R@1_retrieval	0.117	0.889	15.770	2.71e-9	1.65e-30	1	28.144	1.43e-11
MRR_retrieval	0.148	0.862	19.360	3.92e-11	2.24e-31	1	27.059	3.38e-11
nDCG@1_retrieval	0.117	0.889	15.770	2.71e-9	1.65e-30	1	28.144	1.43e-11
Coverage_retrieval	0.271	0.763	33.119	1.35e-17	-3.20e-31	NaN	22.766	1.10e-9
R@1_reranked	0.110	0.896	13.973	2.39e-8	168.678	7.99e-29	10.714	3.69e-5
MRR_reranked	0.175	0.840	21.842	2.30e-12	62.761	1.26e-13	16.315	2.55e-7
nDCG@1_reranked	0.110	0.896	13.973	2.39e-8	168.678	7.99e-29	10.714	3.69e-5
Coverage_reranked	0.271	0.763	33.119	1.35e-17	-3.20e-31	NaN	22.766	1.10e-9
Correctness	0.255	0.775	31.940	4.46e-17	14.614	0.000173	19.648	1.48e-8
Faithfulness	0.357	0.700	24.544	1.13e-13	41.701	7.03e-10	13.627	2.70e-6
Relevance	0.292	0.747	38.249	8.48e-20	11.959	0.000655	18.228	4.93e-8
latency_mean	1.858	0.158	247.010	6.59e-69	4.555	0.034	0.421	0.657
cost_mean	0.000	1.000	45656.247	1.27e-297	0.0063	0.937	0.006	0.994

negative coefficients. Among retrievers, HyPE demonstrates the largest positive gains across all metrics with β for R@1 retrieval of 0.050, whereas hierarchical retrieval shows significant negative effects. Indexing methods (HNSW, IVF, ScaNN) do not exhibit statistically significant influence, indicating that retrieval quality is primarily governed by semantic modeling choices rather than ANN structure.

Table 9 highlights clear trade-offs between system efficiency and answer quality. Inner product similarity significantly improves correctness with β, score of 0.016, and p value of less than 10^{-5} while also reducing computational cost, whereas L2 distance both increases cost and sharply reduces correctness. HyDE substantially increases cost due to additional generation overhead with β value of 2.4, while HyPE provides strong correctness gains with lower cost. The MiniLM reranker consistently improves correctness over BGE, confirming its effectiveness as a lightweight yet accurate reranking model.

VI. RECOMMENDATION

Based on our empirical and statistical findings, we offer the following practical recommendations for dense retrieval-based RAG systems under controlled general-domain settings:

1) Retrieval strategy selection should be prioritized over index optimization, as retrievers explain the largest proportion of variance in both retrieval coverage and generation correctness ($\eta^2 = 0.31$).

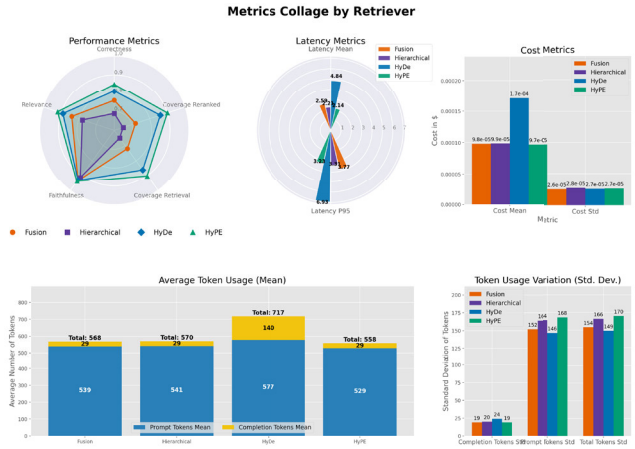

FIGURE 5. Comparison of retrieval strategies.

TABLE 7. Effect size analysis (Eta squared, η^2) quantifying the relative impact of Index type, Retrieval strategy, Reranker, and Similarity Metric on key system performance measures.

Metric	Factor	ETA	Effect Size
Coverage (Retrieval)	Retriever	0.319	Large
Coverage (Reranked)	Retriever	0.319	Large
Correctness	Retriever	0.311	Large
Coverage (Retrieval)	Metric	0.176	Large
Coverage (Reranked)	Metric	0.176	Large
Correctness	Metric	0.156	Large
Coverage (Retrieval)	Reranker	0	Negligible
Coverage (Reranked)	Reranker	0	Negligible
Correctness	Reranker	0.064	Medium
Coverage (Retrieval)	Index	0.003	Negligible
Coverage (Reranked)	Index	0.003	Negligible
Correctness	Index	0.002	Negligible

2) Inner product or cosine similarity should be preferred for dense retrieval, while L2 distance should be avoided due to consistently negative performance effects.
3) Reranking should be applied selectively when downstream correctness and relevance are critical, with lightweight cross-encoders (e.g., MiniLM) offering an optimal accuracy–latency trade-off.
4) Indexing choices should be guided by latency, scalability, and memory constraints rather than expected gains in retrieval effectiveness.
5) Smaller chunks improve precision and correctness, while larger chunks reduce latency and cost, making chunking a context-dependent optimization.

Table 10 shows the summarized recommendation in tabular form.

VII. DISCUSSION

In this study, results were obtained by keeping the system parameters constant for the controlled comparisons. The Chunk size (1,000 characters) and overlap (200 characters) were selected to balance semantic completeness with retrieval granularity. Our results reflect a mid-range configuration that avoids the extremes. Similarly, retrieval depth (top-k) and reranking thresholds were held constant to prevent interaction effects from dominating statistical analysis.

Our findings reveal important interactions between RAG components that are not evident from isolated comparisons. While weaker retrievers limit the impact of reranking regardless of model choice, strong retrievers like HyPE increase their efficacy by providing higher-quality candidate sets. Similarly, similarity metrics affect retrieval strategies differently. For example, when paired with hypothetical embedding-based retrievers, inner-product similarity yields the largest gains, but under hierarchical retrieval, its advantage diminishes. Reranking and similarity metrics should be viewed as conditional amplifiers rather than as separate performance drivers, given these interactions.

Extremely small p-values (e.g., 10^{-9}–10^{-11}) arise due to large sample sizes and low within-group variance; these values are consistent with large observed effect sizes. To improve interpretability, effect sizes (η^2) and 95% confidence intervals are emphasized as primary indicators of practical significance.

The results highlight a clear hierarchy among RAG system components. Across all experiments, retriever choice emerges as the most influential factor, driving large effect sizes for Coverage and Correctness. Similarity metrics represent the second most impactful design dimension. Inner product and cosine similarity consistently improve retrieval and generation quality, while L2 distance degrades performance across almost all evaluated metrics. Reranking contributes meaningfully but selectively. While rerankers have minimal influence on first-stage retrieval metrics, they significantly affect reranked performance and generation correctness. MiniLM demonstrates strong positive effects on reranked R@1 and Correctness, indicating that lightweight cross-encoder rerankers can substantially enhance final answer quality without incurring prohibitive costs. In contrast, index choice (HNSW, IVF, SCANN) shows no statistically significant impact on retrieval quality, with negligible effect sizes across metrics. This suggests that, under sufficient recall conditions, approximate nearest neighbor structures primarily trade off latency rather than effectiveness. Changes in system parameter values may shift absolute metric values, but the relative dominance of retriever choice and similarity metric is expected to remain stable, as confirmed by large effect sizes ($\eta^2 > 0.3$) observed across configurations.

VIII. CONCLUSION

RAG system is a combination of closely associated components working together to solve the problem of LLM hallucination in a large context-aware application. The components (Retrievar, Reranker, Similarity metrics, Data size, Indexer) that form a RAG system vary based on the application. So it is important to understand the impact and the different types of these components. To create a framework and provide tangible recommendations, experiments must be conducted across a variety of configurations. In this study,

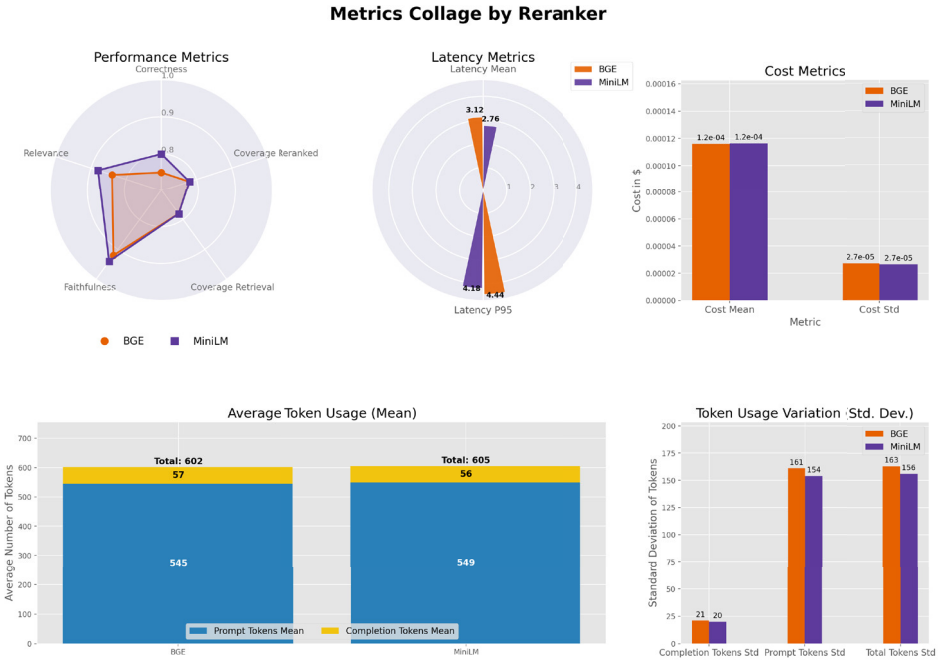

FIGURE 6. Performance comparison between BGE and MiniLM rerankers.

TABLE 8. Multivariate regression analysis showing the influence of indexing method, similarity metric, retrieval strategy, reranker, and dataset size on retrieval and coverage performance.

factor_level	beta	p_value	beta	p_value	beta	p_value	beta	p_value
	R@1_reranked		R@1_retrieval		Coverage_reranked		Coverage_retrieval	
index_hnsw	0.004	0.3976	0.0041	0.538	0.006	0.30198	0.006	0.3020
index_ivf	-0.002	0.6090	-0.0024	0.710	-0.004	0.54078	-0.004	0.5408
index_scann	-0.001	0.7376	-0.0016	0.807	-0.002	0.67337	-0.002	0.6734
metric_cosine	0.015	0.0007	0.0265	7.78e-5	0.023	0.00012	0.023	0.0001
metric_inner_product	0.020	9.55e-6	0.0298	1.01e-5	0.028	4.22e-6	0.028	4.22e-6
metric_l2	-0.036	9.22e-14	-0.0563	2.67e-15	-0.051	1.50e-15	-0.051	1.50e-15
retriever_fusion	-0.013	0.00914	-0.0340	7.65e-6	-0.028	2.38e-5	-0.028	2.38e-5
retriever_hierarchical	-0.044	4.64e-16	-0.0288	0.00013	-0.057	1.90e-15	-0.057	1.90e-15
retriever_hyde	0.021	3.09e-5	0.0125	0.09214	0.032	1.65e-6	0.032	1.65e-6
retriever_hype	0.036	1.11e-11	0.0503	1.13e-10	0.053	8.91e-14	0.053	8.91e-14
reranker_bge	-0.059	6.68e-43	-2.08e-17	0.999	-2.78e-17	0.999	-2.78e-17	0.999
reranker_minilm	0.059	6.68e-43	1.39e-17	0.999	0	1	0	1
size_large	-0.010	0.028	-0.015	0.020	-0.014	0.015	-0.014	0.015
size_medium	-0.001	0.794	-0.002	0.799	-0.002	0.787	-0.002	0.787
size_small	0.011	0.014	0.017	0.010	0.016	0.007	0.016	0.007

we conducted controlled experiments with a variety of RAG components, evaluated performance, and assessed the significance of different RAG components using ANOVA, effect size analysis, and regression modeling. We disentangle the relative importance of pipeline components and quantify their impact on both retrieval and generation performance. Our results demonstrate that retriever and similarity metric choices dominate system behavior, while index structures play a largely infrastructural role. Reranking improves downstream quality but should be applied judiciously. Based on these results, we provide actionable guidance for researchers and practitioners building scalable, high-quality generative systems.

IX. LIMITATIONS

There are several limitations to this study. First, results may not generalize well to multilingual or domain-specific corpora because evaluation is conducted on a general-domain English dataset (SQuAD). Second, to minimize experimental complexity, chunking parameters and retrieval depths are fixed; different configurations may yield different absolute performance values. Third, although controlled analysis is supported by the generation model's (gpt-4o-mini) deliberate lightweight design, stronger generators may yield greater downstream gains. We are using a single embedding model for controlled experiments, which is one of the limitations to full generalization.

TABLE 9. Regression coefficients analyzing the effect of system components on computational cost and answer correctness.

factor_level	beta	p_value	beta	p_value
	Cost		Correctness	
index_hnsw	-1.74e-8	0.670	0.00351	0.2965
index_ivf	3.07e-8	0.453	-0.00203	0.5446
index_scann	-1.33e-8	0.745	-0.00147	0.6606
metric_cosine	-3.70e-8	0.366	0.01242	0.0003
metric_inner_product	-1.09e-7	0.008	0.01593	3.69e-6
metric_l2	1.46e-7	0.000	-0.02836	4.82e-15
retriever_fusion	-7.68e-6	3.57e-221	-0.00839	0.027
retriever_hierarchical	-7.56e-6	8.99e-220	-0.03823	7.32e-20
retriever_hyde	2.36e-5	1.03e-320	0.01599	3.33e-5
retriever_hype	-8.33e-6	2.40e-228	0.03063	4.03e-14
reranker_bge	-8.54e-8	0.0058	-0.01366	1.52e-7
reranker_minilm	8.54e-8	0.0058	0.01366	1.52e-7
size_large	3.95e-7	1.81e-18	-0.00874	0.0098
size_medium	1.33e-7	0.0013	-0.00149	0.6567
size_small	-5.28e-7	2.23e-28	0.01023	0.0026

TABLE 10. Recommendation for rag system based on this study.

Design Dimension	Empirical Finding	Recommendation
Retrieval Strategy	Largest effect size on coverage & correctness	Prioritize retriever selection
Similarity Metric	Inner product & cosine dominate	Avoid L2 distance
Reranking	Improves downstream quality conditionally	Apply selectively
Indexing	Negligible effect on quality	Choose based on latency
Generator	Affects absolute scores only	Use lightweight models for analysis

X. FUTURE WORK

Future work will extend this analysis to multilingual settings, larger-scale corpora, and domain-specific data such as HotpotQA, Natural Questions, and MS MARCO. The work can also be extended by adding more parameters and demonstrating that its findings are transferable to other configurations and datasets. We also plan to study parameter sensitivity more explicitly by varying chunk sizes, overlaps, and retrieval depth. Future experiments will also evaluate whether stronger generation (GPT-4, GPT-5, Claude, etc.) models increase or decrease upstream retrieval effects. This will clarify the separation between retrieval-driven and generation-driven performance gains in RAG systems.

ACKNOWLEDGMENT

Computing resources used in this work were provided by the National Center for High Performance Computing of Türkiye (UHeM). This study is part of a Ph.D. thesis titled "Performance Analysis of Advanced Retrieval-Augmented Generation Applications using Vector Databases, Indexing Algorithms and Distance Metrics" under the supervision of Jawad Rasheed (Cevat Resit).

DECLARATIONS

- Funding Statement: The authors declare that this research received no external funding.
- Conflict of interest/Competing interests: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.
- Ethics Declaration: Not applicable.
- Clinical trial number: not applicable.
- Consent to Participate Declaration: Not applicable.
- Consent for publication: Not applicable.
- Data availability: The dataset (Wikipedia expanded SQuAD for RAG Evaluation - Kaggle) generated for this study is publicly available at: https://doi.org/10.34740/KAGGLE/DSV/14354277 and https://www.kaggle.com/dsv/14354277

REFERENCES

[1] R. Nogueira, W. Yang, J. Lin, and K. Cho, "Document expansion by query prediction," 2019, *arXiv:1904.08375*.

[2] Y. Qu, Y. Ding, J. Liu, K. Liu, R. Ren, W. X. Zhao, D. Dong, H. Wu, and H. Wang, "RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2021, pp. 5835–5847.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol., (Long Short Papers)*, vol. 1, 2019, pp. 4171–4186.

[4] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, and Y. Zhang, "A survey on large language model (LLM) security and privacy: The good, the bad, and the ugly," *High-Confidence Comput.*, vol. 4, no. 2, Jun. 2024, Art. no. 100211.

[5] J. Liu, Y. Tao, F. Wang, H. Li, and X. Qin, "SiQA: A large multi-modal question answering model for structured images based on RAG," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2025, pp. 1–5.

[6] A. Yates, R. Nogueira, and J. Lin, "Pretrained transformers for text ranking: BERT and beyond," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, Mar. 2021, pp. 1154–1156.

[7] X. Li, Y. Bai, B. Jin, F. Zhu, L. Pan, and Y. Cao, "Long context vs. RAG: Strategies for processing long documents in LLMs," in *Proc. 48th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2025, pp. 4110–4113.

[8] Z. Ji, T. Yu, Y. Xu, N. Lee, E. Ishii, and P. Fung, "Towards mitigating LLM hallucination via self reflection," in *Proc. Findings Assoc. Comput. Linguistics: EMNLP*, 2023, pp. 1827–1843.

[9] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-T. Yih, and T. Rocktäschel, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 9459–9474.

[10] A. J. Oche, A. G. Folashade, T. Ghosal, and A. Biswas, "A systematic review of key retrieval-augmented generation (RAG) systems: Progress, gaps, and future directions," 2025, *arXiv:2507.18910*.

[11] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," 2023, *arXiv:2312.10997*.

[12] T. Dhabalia, S. Bhate, K. Singh, B. Cerejo, and D. Bhagat, "A comparative study of RAG and fine-tuned transformer models for domain-specific chatbots," in *Proc. Int. Conf. Intell. Cloud Comput. (ICoICC)*, May 2025, pp. 1–6.

[13] M. Shen, M. Umar, K. Maeng, G. Edward Suh, and U. Gupta, "Towards understanding systems trade-offs in retrieval-augmented generation model inference," 2024, *arXiv:2412.11854*.

[14] N. Ampazis, "Improving rag quality for large language models with topic-enhanced reranking," in *Proc. IFIP Int. Conf. Artif. Intell. Appl. Innov.* Cham, Switzerland: Springer, 2024, pp. 74–87.

[15] J. Dong, B. Fatemi, B. Perozzi, L. F. Yang, and A. Tsitsulin, "Don't forget to connect! Improving RAG with graph-based reranking," 2024, *arXiv:2405.18414*.

[16] D. Vake, J. Vičič, and A. Tošić, "Bridging the question-answer gap in retrieval-augmented generation: Hypothetical prompt embeddings," *IEEE Access*, vol. 13, pp. 129952–129961, 2025.
[17] M. Arslan, H. Ghanem, S. Munawar, and C. Cruz, "A survey on RAG with LLMs," *Proc. Comput. Sci.*, vol. 246, pp. 3781–3790, Jan. 2024.
[18] Q. M. Ilyas and S. Aziz, "Enhancing the RAG pipeline through advanced optimization techniques," in *Generative AI Foundations, Developments, and Applications*. IGI Global, 2025, pp. 59–80.
[19] F. Brontes, J. Genesis, Z. Noa, and S. Nymphodoros, "Learning to retrieve, generate, and compress: A unified view of efficient RAG," *Preprints*, Aug. 2025, doi: 10.20944/preprints202508.1211.v1.
[20] N. Zhang, P. K. Choubey, A. Fabbri, G. Bernadett-Shapiro, R. Zhang, P. Mitra, C. Xiong, and C.-S. Wu, "SiReRAG: Indexing similar and related information for multihop reasoning," 2024, *arXiv:2412.06206*.
[21] K. R. Ong and W. P. Wong, "Optimizing information retrieval in RAG through intelligent reranking and follow-up query predictions," *Res. Square*, Sep. 2025. [Online]. Available: https://doi.org/10.21203/rs.3.rs-7506627/v1
[22] S. Dadas and M. Grebowiec, "Assessing generalization capability of text ranking models in Polish," in *Proc. Int. Conf. Artif. Intell. Soft Comput.* Cham, Switzerland: Springer, 2024, pp. 37–49.
[23] D. Tanyildiz, S. Ayvaz, and M. F. Amasyali, "Enhancing retrieval-augmented generation accuracy with dynamic chunking and optimized vector search," *Orclever Proc. Res. Develop.*, vol. 5, no. 1, pp. 215–225, Dec. 2024.
[24] G. de Souza P. Moreira, R. Ak, B. Schifferer, M. Xu, R. Osmulski, and E. Oldridge, "Enhancing Q&A text retrieval with ranking models: Benchmarking, fine-tuning and deploying rerankers for RAG," 2024, *arXiv:2409.07691*.
[25] A. K. Shahade and P. V. Deshmukh, "Enhancing natural language processing: A comprehensive review of retrieval augmented generation," in *Proc. 4th Int. Conf. Sustain. Expert Syst. (ICSES)*, Oct. 2024, pp. 609–611.
[26] X. Yang, K. Sun, H. Xin, Y. Sun, N. Bhalla, X. Chen, S. Choudhary, R. D. Gui, Z. W. Jiang, and Z. Jiang, "CRAG-comprehensive rag benchmark," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 37, 2024, pp. 10470–10490.
[27] S. Wang, J. Tan, Z. Dou, and J.-R. Wen, "OmniEval: An omnidirectional and automatic RAG evaluation benchmark in financial domain," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2025, pp. 5737–5762.
[28] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," 2016, *arXiv:1606.05250*.
[29] H. Elkiran, "Wikipedia expanded squad for rag evaluation," Kaggle, Istanbul, Dec. 2025, doi: 10.34740/KAGGLE/DSV/14354277.
[30] H. Brådland, M. Goodwin, P.-A. Andersen, A. S. Nossum, and A. Gupta, "A new HOPE: Domain-agnostic automatic evaluation of text chunking," in *Proc. 48th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2025, pp. 170–179.
[31] Z. Zhong, H. Liu, X. Cui, X. Zhang, and Z. Qin, "Mix-of-granularity: Optimize the chunking granularity for retrieval-augmented generation," in *Proc. 31st Int. Conf. Comput. Linguistics*, 2025, pp. 5756–5774.
[32] S. Wu, J. Xie, J. Chen, T. Zhu, K. Zhang, and Y. Xiao, "How easily do irrelevant inputs skew the responses of large language models?" 2024, *arXiv:2404.03302*.
[33] J. Wang, "Milvus: A purpose-built vector data management system," in *Proc. Int. Conf. Manage. Data*, Jun. 2021, pp. 2614–2627.
[34] A. Andoni, P. Indyk, and I. Razenshteyn, "Approximate nearest neighbor search in high dimensions," in *Proc. Int. Congr. Mathematicians (ICM)*, May 2019, pp. 3287–3318.
[35] K. Juvekar and A. Purwar, "COS-mix: Cosine similarity and distance fusion for improved information retrieval," 2024, *arXiv:2406.00638*.
[36] J. Huo, "Tool-integrated RAG framework for Scottish Gazetteer retrieval and geographic visualization," School GeoSciences, Univ. Edinburgh, Edinburgh, 2025, doi: 10.7488/era/6741.
[37] A. Khaledian, A. Ghadiridehkordi, and N. Khaledian, "PCA-RAG: Principal component analysis for efficient retrieval-augmented generation," 2025, *arXiv:2504.08386*.
[38] Y. Sun, R. Zhang, R. Meng, L. Lian, H. Wang, and X. Quan, "Fusion-based retrieval-augmented generation for complex question answering with LLMs," in *Proc. 8th Int. Conf. Comput. Inf. Sci. Appl. Technol. (CISAT)*, Jul. 2025, pp. 116–120.
[39] A. Sorstkins, "Assessing RAG and HyDE on 1B vs. 4B-parameter gemma LLMs for personal assistants integretion," 2025, *arXiv:2506.21568*.

HARUN ELKIRAN (Graduate Student Member, IEEE) received the M.S. degree in computer science and engineering from Istanbul Sabahattin Zaim University, Istanbul, Türkiye, where he is currently pursuing the Ph.D. degree in computer science and engineering, under the supervision of Dr. Jawad Rasheed. His research interests include RAG, LLM, deep learning, and database systems and management.

JAWAD RASHEED (Member, IEEE) received the B.S. degree in telecommunication engineering from the National University of Computer and Emerging Sciences, Pakistan, and the M.S. degree in electrical and electronics engineering and the Ph.D. degree in computer science and engineering in Türkiye.

He is currently an Associate Professor with the Department of Computer Engineering, Istanbul Sabahattin Zaim University, Türkiye. He is also a Senior Researcher with Istanbul Medipol University and Instanbul Nisantasi University, Türkiye. He is the author/co-author of more than 90 papers published in reputable journals and highly ranked conferences. His research interests include artificial intelligence and image processing. pattern recognition, the IoT, LLM, RAG, blockchain, and data analytics. He was a Gold Medalist and was awarded the Academic Excellence Award for securing straight A's in O-Level exams held by Cambridge University. Later, he also received a prestigious Doctorate and Research Scholarship for his Ph.D. studies (for three years). He serves as an Editor/Guest Editor for several reputed journals, including *BMC Infectious Disease*, *PLoS ONE*, *International Journal of Computational Intelligence Systems*, *Discover Artificial Intelligence*, and *International Journal of Intelligent Transportation Systems Research*.