IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 21 November 2025, accepted 27 November 2025, date of publication 22 December 2025,
date of current version 29 December 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3646665

RESEARCH ARTICLE

EvaRAG: Evaluating Advanced RAG Techniques With Indexing and Distance Metrics

HARUN ELKIRAN[1], (Member, IEEE),
AND JAWAD RASHEED[1,2,3,4], (Member, IEEE)
[1]Department of Computer Engineering, Istanbul Sabahattin Zaim University, 34303 Istanbul, Türkiye
[2]Department of Software Engineering, Istanbul Nisantasi University, 34398 Istanbul, Türkiye
[3]Research Institute, Istanbul Medipol University, 34810 Istanbul, Türkiye
[4]Applied Science Research Center, Applied Science Private University, Amman, Jordan

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

ABSTRACT Retrieval Augmented Generation (RAG) has emerged as a powerful paradigm for enhancing large language models (LLMs) with external knowledge. Yet, the performance of RAG pipelines is susceptible to design choices across retrieval, similarity metrics, indexing, and reranking. Despite growing adoption, little systematic work has explored the trade-offs between retrieval quality, semantic accuracy, computational efficiency, and cost in RAG systems. This study addresses this gap by conducting a comprehensive evaluation of RAG configurations across multiple dimensions. We propose a benchmarking framework that systematically varies retrievers (Fusion, HyDe, Hierarchical, SCaNN), indexing methods (HNSW, IVF, Flat), similarity metrics (Cosine, Inner Product, L2), and rerankers (BGE, minilm) over datasets of three scales (small, medium, and large). Performance is assessed through coverage, recall, MRR, and nDCG, while semantic quality is measured using correctness, faithfulness, and relevance. Efficiency is quantified via latency, throughput, and computational cost. Our experiments reveal that HNSW–IP–Fusion–minilm achieves the strongest semantic performance, with Coverage Retrieval of 0.942, Correctness of 0.909, and Faithfulness of 0.970, making it ideal for accuracy-critical tasks. Conversely, IVF–L2–Hierarchical demonstrates the lowest latency (1.736 ns) and cost, making it suitable for real-time deployments. Reranker analysis shows modest but consistent gains for minilm over BGE, while HyDe excels in precision at the expense of efficiency. Notably, no single configuration dominates; optimal designs depend on the application's needs, whether it is maximizing semantic accuracy, minimizing latency, or striking a balance between the two. By demonstrating concrete trade-offs, this work provides a practical foundation for scaling RAG pipelines across diverse domains, including information retrieval, enterprise search, and knowledge-intensive reasoning.

INDEX TERMS Data retrieval, large language model, natural language processing, question answering systems, RAG.

I. INTRODUCTION

Recent breakthroughs in artificial intelligence and natural language processing have led to the development of powerful large language models (LLMs), such as the Generative Pre-trained Transformer (GPT). The rapid progress of LLMs can be attributed to improvements in deep learning techniques, the development of large-scale transformers, and the availability of massive datasets. Models such as GPT-4 [1] and Llama 2 [2] excel in various tasks and domains, often without prompts.

These models have significant potential in various domains, including coding, medicine, law, agriculture, and psychology, and are approaching human-level knowledge, [3], [4], [5], [6]. Although LLMs have extensive pre-training knowledge, their lack of customized domain-specific understanding or knowledge of recent events can lead to outdated or unfounded responses in real-world applications,

The associate editor coordinating the review of this manuscript and approving it for publication was Maria Chiara Caschera.

also known as hallucinations [7], [8], [9], [10]. Unreliability is a significant barrier to the safe adoption of LLM-based systems for essential business applications, due to user trust issues stemming from hallucinations.

Retrieval Augmented Generation (RAG) has recently gained adoption in question-answering systems because it enhances query context and improves semantic grounding [11]. Text retrieval is crucial for various information retrieval applications, including search, (Q&A), and different types of recommendation systems. RAG systems [12], [13] retrieve text to provide scope to LLMs. RAG pipelines use approximate nearest-neighbor (ANN) search methods to retrieve relevant documents from large databases. The LLM uses the retrieved text chunk to provide accurate, timely responses.

The retrieval stage is a critical component of RAG pipelines, as it relies on similarity measurements and efficient indexing systems. Semantic closeness between embeddings is measured through metrics, such as cosine similarity, Euclidean distance, and inner product. To scale this process, approximate nearest neighbor search (ANNS) [14] methods use indexing techniques [13] such as Hierarchical Navigable Small World (HNSW), ScaNN, and Inverted File Index (IVF). These indices enable efficient retrieval from large embedding collections by organizing vectors into graph- or clustering-based structures. Beyond initial retrieval, rerankers [15] like BGE and MiniLM enhance candidate results by utilizing cross-encoder models, thereby improving contextual alignment between the query and retrieved passages.

While RAG offers a promising direction for mitigating hallucinations, existing studies typically evaluate RAG pipelines under limited conditions. Most prior work focuses on a single retriever-reranker configuration or on small-scale datasets, making it challenging to compare design choices systematically. There remains a lack of comprehensive frameworks that explore the whole design space of chunking, embeddings, retrievers, rerankers, and efficiency trade-offs (accuracy, latency, and cost). To address this issue, this study conducts a comparative analysis of various indexes, distance metrics, dataset sizes, retrievers, and rerankers, providing conclusive results on which configuration performs better under specific conditions. The following are the key contributions of this study.

1) Presenting EVARAG, a reproducible RAG benchmarking pipeline that integrates dataset preparation, chunking, embeddings, indexes (HNSW, ScaNN, and IVF), retrievers (Fusion, Hierarchical, and HyDe), and rerankers (BGE, MiniLM).
2) Conducted 162 experiments across multiple dataset scales, enabling controlled comparisons of retrieval effectiveness, generative quality, and efficiency trade-offs.
3) Revealing the strengths and limitations of indexes, distance/similarity metrics, retrievers, and rerankers, as well as the trade-offs between accuracy, latency, and cost.

The remainder of this paper is structured as follows. Section II reviews related work on RAG and evaluation methodologies. Section III describes the EVARAG methodology in detail. Section IV presents experimental results across all configurations. Section V discusses key findings and implications. Finally, Section VI concludes the paper with directions for future research.

II. LITERATURE REVIEW

RAG integrates retrieval and generation techniques to enhance language modelling tasks. The RAG pipeline has two major steps: retrieving relevant information and generating contextually informed text. Table 1 summarizes key RAG studies, highlighting the retrievers, rerankers, indexing methods, and distance/similarity metrics employed.

RAG systems use a range of retrieval methods, including semantic search and FAISS-based similarity matching, to efficiently identify relevant information [16]. To retrieve the most pertinent text, the choice of index plays a central role. Approaches such as HNSW [17], ScaNN [14], and IVF [18] enable scalable ANNs through their efficient indexing schemes. These approaches strike a balance between recall and latency. Similarly, FAISS has been shown to achieve strong performance in high-dimensional vector search [19]. These methods reduce search complexity but often trade recall for efficiency, which impacts the quality of downstream text generation, a key limitation of these systems.

RAG performance is strongly influenced by how documents are chunked, retrieved, and filtered before generating responses. Effective chunking strategies, such as those evaluated by holistic frameworks like HOPE [20], can boost factual accuracy and coherence. Meanwhile, semantically guided chunking using LLMs to produce richer, more meaningful text segments and metadata [21]. Retrieval quality also remains a core challenge. Techniques like W-RAG [22] leverage signals from LLM behavior to improve retriever training, achieving performance close to that of human-labeled data. For filtering noisy or irrelevant retrievals, multi-agent approaches like MAIN-RAG [23] tap into LLM consensus all without extra training. By rearranging candidate documents according to semantic relevance, reranking improves the initial retrieval results. Cross-encoders and lightweight transformer-based models, such as BGE [24] and MiniLM [25], are widely adopted as rerankers. Reranking has been shown to improve the contextual accuracy of retrieved outputs [19]. Moreover, hybrid systems that combine multiple retrievers with rerankers often outperform single-method approaches, particularly for complex or deep logic queries [24]. Other strategies, such as HyDE, further enrich retrieval by generating hypothetical documents to expand the search space.

Evaluating the effectiveness of retrievers relies on similarity measures between query and document embeddings. Standard metrics include cosine similarity and Euclidean distance,

with cosine similarity demonstrating superior performance in most RAG applications [24]. Embedding-based scoring methods, such as BERT embedding similarity, also provide strong alignment with semantic meaning. Recently, the Overall Performance Index (OPI) was proposed, combining logical correctness and embedding similarity into a unified measure for holistic RAG evaluation [24].

Several benchmarks have been introduced to assess retrieval and generation. KILT [12] integrates knowledge-intensive NLP tasks, whereas domain-specific studies in medicine [4]. More recent evaluations focus on hallucination mitigation and factual consistency [8]. Despite these contributions, most benchmarks evaluate either retrieval or generation in isolation, with limited insight into trade-offs across system configurations. This leaves practitioners with little guidance on the interplay between pipeline components, such as retrievers, indexers, and rerankers, and distance/similarity metrics and dataset size. EVARAG closes this gap by providing a unified, reproducible framework for systematically comparing RAG configurations. By varying indexes, retrieval and reranking strategies, and measuring both effectiveness and efficiency metrics, EVARAG provides actionable, holistic insights into the trade-offs and best-fit configurations.

III. METHODOLOGY

The proposed methodology for EVARAG follows a structured RAG pipeline, as illustrated in Fig. 1. The workflow integrates datasets, embedding modelling, indexing, retrieval, reranking, and evaluation. The entire process is designed to be modular, reproducible, and extensible, enabling comprehensive benchmarking under controlled conditions. The methodology is described in detail below.

A. DATASETS

The first step in the EVARAG is data collection. In this study, we used the Stanford Question Answering Dataset (SQuAD) [31]. It is widely used as a benchmark for question-answering tasks. SQuAD consists of more than 100,000 question-answer pairs generated by human annotators from Wikipedia articles. Each instance in the dataset includes a context passage, a question, and one or more ground truth answers. This dataset is particularly well-suited for evaluating RAG systems where both context retrieval and answer generation quality are important. In this study, to capture the impact of dataset scale on RAG performance, three dataset configurations are used: Small Dataset of 10,000 documents for fast baseline experimentation and debugging, a Medium Dataset of 30,000 documents to examine performance trends as the retrieval space grows, and a Large Dataset of 100,000 documents to approximate production level scenarios, stressing both indexing efficiency and retrieval accuracy.

B. CHUNKING

Document chunking is a crucial preprocessing step that enables efficient and scalable retrieval within the RAG pipeline. In chunking, instead of treating each document as a single large unit, the corpus is segmented into overlapping chunks. Chunking allows fine-grained retrieval. It preserves contextual continuity. Each chunk is uniquely identified by the tuple (`doc_id`, `chunk_id`), ensuring precise traceability back to the original document. It is used in ground truth evaluation. In this study, chunks were created with a fixed window size of 1,000 characters and an overlap of 200 characters. This ratio strikes a balance between granularity and contextual completeness. The overlap set in this study ensures that information appearing near chunk boundaries is not lost. It preserves the narrative flow and semantic continuity across adjacent chunks. Chunk overlap is significant for tasks involving question answering or knowledge-intensive reasoning, where critical context may span multiple segments. This strategy of carefully selecting an overlap value improves retrieval precision by narrowing the search to contextually relevant passages while maintaining high recall through overlap, ultimately leading to more accurate, contextually grounded responses. The resulting chunked representation facilitates efficient embedding generation, as each chunk can be independently encoded into a vector space for downstream semantic search.

C. EMBEDDING MODELING

Following chunk creation, each segment is transformed into a dense vector embedding to enable semantic retrieval. This vector representation enables semantic search and similarity-based retrieval. For this purpose, we have used OpenAI's `text-embedding-3-large` model, one of the most advanced embedding models for capturing rich semantic and contextual information from natural language text. It produces dense embeddings with a maximum dimensionality of $d = 3072$.

Mathematically, $\mathcal{C}(D) = \{C_1, C_2, \ldots, C_n\}$ denotes the set of chunks generated from a document D. Each chunk C_i is mapped into a continuous vector space using the embedding function $f_\theta(\cdot)$.

$$\mathbf{e}_i = f_\theta(C_i), \quad \mathbf{e}_i \in \mathbb{R}^d, \quad d = 3072. \tag{1}$$

Equation (1) produces a dense embedding vector $\mathbf{e}_i$ for each chunk, resulting in an embedding matrix shown by equation 2

$$\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n]^\top \in \mathbb{R}^{n \times d}, \tag{2}$$

Together, Equations (1) and (2) define the semantic representation layer of the RAG pipeline. The resulting matrix $\mathbf{E}$ serves as the foundation for indexing and similarity search.

TABLE 1. Summary of recent literature on Retrieval-Augmented Generation (RAG).

Citation	Focus / Contribution	Retrievers / Indexing	Rerankers	Key Metrics / Results
[27]	Comprehensive review of RAG architecture and components	General retrievers	General rerankers	Foundational principles; integration of retrieval into generation
[16]	Job recommendation system using RAG + NLP	FAISS similarity search	–	+25.8% Precision@10; +41.2% job applications
[20]	Dynamic chunking and optimized vector search for RAG	FAISS-HNSW	Cross-encoder reranker	Improved response fidelity; index choice significantly impacts accuracy
[28]	Optimal search for RAG; ANN accuracy vs. speed trade-offs	ANN retrievers	–	Correctness plateaus at 5–10 docs; more gold docs improve QA scores
[25]	Evaluation of retrievers for deep-logic questions	DPS, EDI	–	OPI correlates with extrinsic eval; cosine similarity retriever strongest
[29]	Case study: domain-specific QA (Pittsburgh, CMU)	BM25, FAISS	Reranker (not specified)	F1 = 42.21 (vs. 5.45 baseline)
[30]	Graph-based reranking (G-RAG) using AMR connections	General retrievers	Graph-based reranker	G-RAG outperforms PaLM 2; reranking shown crucial
[26]	Multi-way recall fusion reranking with Tensor + ColBERT	General retrievers	ColBERT, Tensor-based	Improved recall and precision; balanced compute vs. speed
[17]	Comparative analysis of real-world retrieval systems (AWS, GCP)	Multiple retrieval systems	–	8 retrievers tested; graph search best on RobustQA and speed
[31]	RAG Playground: evaluation framework for retrieval + prompts	Naive vector search, hybrid search	Hybrid reranking	Pass rate 72.7%; Qwen 2.5 > Llama 3.1 in accuracy

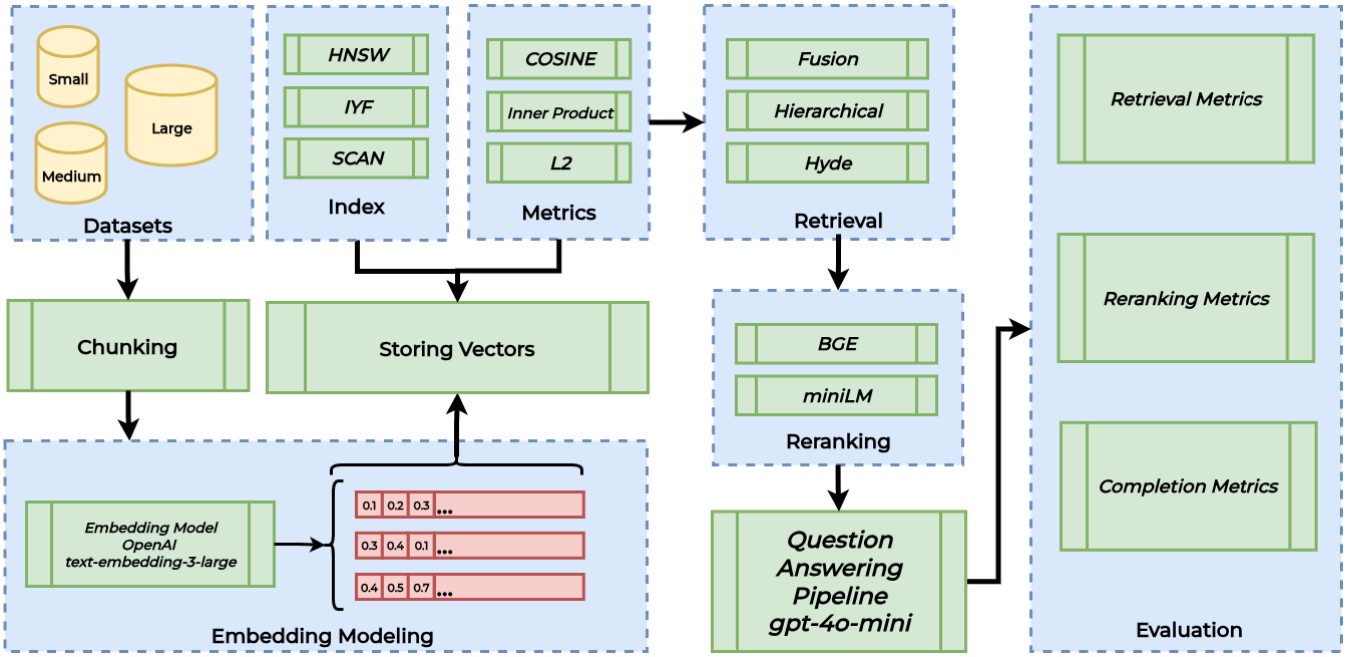

FIGURE 1. Illustration of the complete experimental pipeline, including dataset selection (large, medium, small), indexing strategies (HNSW, IVF, ScaNN), distance metrics (cosine, inner product, L2), retrieval strategies (fusion, hierarchical, HyDE), and reranking (BGE, MiniLM) used to generate and evaluate results.

D. STORING VECTORS

Once the embeddings are generated, the next step is to store them in a vector database to enable scalable, low-latency, similarity-based retrieval. Each embedding $\mathbf{e}_i$ from Equation (1) is stored together with its corresponding `doc_id` and `chunk_id`. This storage mechanism ensures precise traceability back to the original document. These metadata connections allow downstream evaluation. This allows retrieved results to be directly compared with ground-truth answers for recall and relevance analysis. In this study, we use Milvus, a high-performance, open source vector database specifically designed for large-scale similarity search [25]. Milvus supports both dense and hybrid (dense + sparse) embeddings and is optimized for real-time ANN queries.

E. INDEXING STRATEGIES

To enable efficient similarity search at scale, we construct ANN indexes for each dataset configuration (10k, 30k, 100k embeddings). The choice of indexing strategy has a

significant impact on both latency and recall, especially in large embedding spaces where exhaustive search is computationally expensive. Milvus supports several ANN algorithms, and we have experimented with three widely adopted methods. Let's discuss each one of them.

1) HNSW (HIERARCHICAL NAVIGABLE SMALL WORLD)

HNSW is a graph-based ANN algorithm that organizes the embedding set $\mathcal{E} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n\}$ into a hierarchy of proximity graphs. Each layer $\ell \in \{1, \ldots, L\}$ contains a subset of nodes, with edges connecting each vector $\mathbf{e}_i$ to its M nearest neighbors according to the chosen distance metric (e.g., cosine or L2). At query time, the search procedure performs greedy routing through graph G, starting from an entry point in the top layer and iteratively descending through layers to reach the base layer, where a local neighborhood search is conducted. The complexity of HNSW is shown by equation 3

$$T_{\text{search}} = \mathcal{O}(\log n) \tag{3}$$

Equation 3 shows that HNSW is highly efficient for large-scale, high-dimensional embeddings. The graph-based structure allows HNSW to avoid exhaustive scanning, significantly reducing query latency while maintaining near-exact recall.

2) IVF (INVERTED FILE INDEX)

IVF partitions the embedding space into k coarse clusters by performing k mean clustering over all embeddings during the indexing stage. This process assigns each embedding vector $\mathbf{e}_i$ to its nearest centroid $\boldsymbol{\mu}_{c(i)}$, minimizing the within-cluster variance. Equation (4) ensures that clusters are formed optimally, supporting efficient coarse-to-fine retrieval.

$$\min_{\{\boldsymbol{\mu}_j\}_{j=1}^{k}} \sum_{i=1}^{n} \left\|\mathbf{e}_i - \boldsymbol{\mu}_{c(i)}\right\|_2^2. \tag{4}$$

At query time, only the n_{probe} clusters whose centroids are closest to the query embedding are searched, significantly reducing the candidate set size. As a result, IVF achieves a tunable balance between search speed and recall.

3) SCANN (SCALABLE NEAREST NEIGHBORS)

ScaNN is a state-of-the-art ANN method that combines tree partitioning with vector quantization to accelerate nearest neighbor search. During indexing, the embedding space is partitioned into clusters, and each embedding vector $\mathbf{e}_i$ is stored as a quantized code $\hat{\mathbf{e}}_i$ that minimizes the reconstruction error. In ScaNN, the quantized representation preserves semantic proximity, enabling efficient approximate retrieval as shown in Equation (5). ScaNN keeps the recall close to exact search.

$$\hat{\mathbf{e}}_i = \arg\min_{\mathbf{q} \in \mathcal{Q}} \left\|\mathbf{e}_i - \mathbf{q}\right\|_2^2, \tag{5}$$

where $\mathcal{Q}$ is the codebook of quantized vectors. At query time, ScaNN uses a two-stage search: first, a coarse search over partition centroids, then a re-ranking of the top k candidates using the exact distance between $\mathbf{q}$ and $\mathbf{e}_i$.

F. SIMILARITY METRICS

The choice of similarity metric plays a critical role in determining which vectors are considered nearest neighbors. To assess the relevance of candidate chunks to a given query, a similarity function $s(\mathbf{q}, \mathbf{e}_i)$ is used to compare the query embedding $\mathbf{q} \in \mathbb{R}^d$ with each candidate embedding $\mathbf{e}_i \in \mathbb{R}^d$. In this work, we explore three widely used formulations for $s(\cdot, \cdot)$ cosine similarity, inner product, and L2 distance, each offering a different notion of closeness in the embedding space. These three metrics are commonly used and are supported by Milvus.

1) COSINE SIMILARITY

Cosine similarity measures the angular closeness between two vectors, capturing the degree to which they point in the same direction, irrespective of their magnitudes. Formally, for a query embedding $\mathbf{q} \in \mathbb{R}^d$ and a candidate embedding $\mathbf{e}_i \in \mathbb{R}^d$, the cosine similarity score is defined by equation 6

$$s_{\cos}(\mathbf{q}, \mathbf{e}_i) = \frac{\mathbf{q} \cdot \mathbf{e}_i}{\|\mathbf{q}\|_2 \|\mathbf{e}_i\|_2}. \tag{6}$$

Equation (6) produces values in the range $[-1, 1]$, where 1 indicates perfect directional alignment and -1 indicates complete opposition. For most contextual embeddings derived from modern language models, values typically range between 0 and 1.

2) INNER PRODUCT (DOT PRODUCT SIMILARITY)

Inner product similarity directly measures the alignment and magnitude between a query embedding $\mathbf{q} \in \mathbb{R}^d$ and a candidate embedding $\mathbf{e}_i \in \mathbb{R}^d$. It is shown by equation 7

$$s_{\text{ip}}(\mathbf{q}, \mathbf{e}_i) = \mathbf{q} \cdot \mathbf{e}_i. \tag{7}$$

Equation (7) produces a scalar value that increases with both directional alignment and vector magnitude, making it particularly suitable for ranking-based retrieval systems where larger dot products correspond to higher similarity or relevance.

3) L2 DISTANCE (EUCLIDEAN DISTANCE)

L2 distance, also known as Euclidean distance, measures the absolute geometric separation between a query embedding $\mathbf{q} \in \mathbb{R}^d$ and a candidate embedding $\mathbf{e}_i \in \mathbb{R}^d$ using equation 8

$$d_{L2}(\mathbf{q}, \mathbf{e}_i) = \|\mathbf{q} - \mathbf{e}_i\|_2 = \sqrt{\sum_{j=1}^{d} (q_j - e_{i,j})^2}. \tag{8}$$

Unlike other similarity-based metrics, a lower L2 distance indicates closer neighbors. This metric is sensitive to vector magnitude, which can be desirable when absolute position in the embedding space encodes semantic confidence.

G. RETRIEVAL STRATEGIES

After storing the embeddings in the vector database, the next step is to develop retrieval strategies that efficiently fetch relevant chunks. In this study, we evaluate three complementary retrieval paradigms: Fusion Retrieval, Hierarchical Retrieval, and Hypothetical Document Embeddings (HyDE). Each retrieval strategy is implemented on top of the same Milvus database, ensuring that performance differences are attributable to the retrieval approach rather than the database configuration.

1) FUSION RETRIEVAL

Fusion retrieval is a rank-aggregation strategy that combines the results of multiple queries or retrieval configurations. Instead of relying on a single retrieval output, we compute ranked lists from different similarity metrics (Cosine, Inner Product, L2) and merge them using rank fusion techniques such as Reciprocal Rank Fusion (RRF). The RRF score of a document d across k rank lists is given by equation 9

$$\mathrm{RRF}(d) = \sum_{i=1}^{k} \frac{1}{c + r_i(d)} \tag{9}$$

In equation 9 $r_i(d)$ is the rank position of document d in the i^{th} list, and c is a constant (typically $c = 60$) used to smooth the contribution of lower-ranked results. This ensures that documents appearing consistently in the top positions across retrieval methods receive higher final scores.

2) HIERARCHICAL RETRIEVAL

Hierarchical retrieval decomposes the search process into multiple stages to improve efficiency and relevance. In the first stage, we perform a coarse-grained retrieval using a fast, low-dimensional embedding space to obtain a candidate set $\mathcal{C}$. In the second stage, a fine-grained reranking step is applied using the original high-dimensional embeddings using a scoring function. Mathematically, let q be the query embedding, $\mathbf{E}$ be the set of all document embeddings, and $\mathcal{C}$ be the candidate set retrieved in stage one, it is represented by equation 10.

$$\mathcal{C} = \mathrm{TopK}\big(\mathrm{ANN}_{\mathrm{coarse}}(q, \mathbf{E})\big) \tag{10}$$

Then, the final ranked list $\mathcal{R}$ is produced by re-scoring $\mathcal{C}$ using a more accurate similarity function S, which is represented by equation 11.

$$\mathcal{R} = \mathrm{Sort}\big(\{(d, S(q, d)) \mid d \in \mathcal{C}\}\big) \tag{11}$$

This two-stage approach balances computational cost and accuracy, as expensive reranking is only applied to a small candidate set.

3) HYDE RETRIEVAL

Hypothetical Document Embeddings (HyDE) retrieval is a generative-augmented retrieval technique in which the query is expanded by generating a synthetic "hypothetical answer" before embedding. The process consists of three steps:

1) Generate a pseudo answer $\hat{a}$ using a generative model such as GPT.
2) Compute the embedding $\mathbf{e}_{\hat{a}}$ of $\hat{a}$ using the same embedding model used for document chunks.
3) Perform retrieval using $\mathbf{e}_{\hat{a}}$ instead of $\mathbf{e}_q$, thereby pulling semantically richer documents that align with the generated context.

HyDE is particularly effective when queries are sparse or underspecified, as the generative step introduces additional context that helps match relevant chunks.

H. RERANKING

After initial retrieval, reranking is applied to refine the candidate list and improve the relevance of retrieved chunks for downstream generative tasks. Reranking leverages more computationally expensive, context-aware models that can consider pairwise interactions between the query and each candidate document. In this study, we experiment with two state-of-the-art reranking models: BGE and MiniLM Cross-Encoder.

1) BGE (BI-ENCODER) RERANKING

BGE is a bi-encoder architecture that maps both the query and candidate documents independently into a shared embedding space, enabling efficient similarity computation. Let q be the query and d_i a candidate chunk, with embeddings $\mathbf{e}_q = f_{\theta_q}(q)$ and $\mathbf{e}_{d_i} = f_{\theta_d}(d_i)$. The similarity score is computed using equation 12

$$s_{\mathrm{BGE}}(q, d_i) = \cos(\mathbf{e}_q, \mathbf{e}_{d_i}) = \frac{\mathbf{e}_q \cdot \mathbf{e}_{d_i}}{\|\mathbf{e}_q\|_2 \|\mathbf{e}_{d_i}\|_2}. \tag{12}$$

Because embeddings are precomputed for the candidate documents, scoring can be performed efficiently using vector operations over the top k retrieved candidates. BGE excels at capturing semantic similarity while remaining computationally feasible for large candidate sets.

2) MINILM CROSS-ENCODER

MiniLM Cross Encoder is a transformer-based model that jointly encodes the query and candidate document, allowing full attention across both sequences. Unlike bi-encoders, this approach models fine-grained interactions between query tokens and document tokens. For query q and document d_i, the cross encoder produces a scalar relevance score using equation 13

$$s_{\mathrm{CE}}(q, d_i) = \mathrm{CrossEncoder}(q, d_i; \theta_{\mathrm{CE}}), \tag{13}$$

In equation 13 θ_{CE} are the parameters of the MiniLM model. The cross encoder directly outputs a relevance probability or score, which is used to rerank the top k candidates retrieved by the ANN index. Although Cross-Encoders provide superior accuracy by modeling token-level interactions, they are computationally more expensive than bi-encoder methods, making them suitable primarily for reranking a small subset of candidate documents rather than the entire corpus.

I. QUESTION ANSWERING PIPELINE

After the final reranking step, the top k most relevant document chunks are selected to serve as contextual evidence for the language model. Let $\mathcal{C}_{\text{top}} = \{C_1, C_2, \ldots, C_k\}$ denote the set of these top-ranked chunks returned by the retrieval and reranking stages. The user query Q is then combined with these chunks to form a structured input prompt for the language model using equation 14.

$$\text{Input} = \text{FormatPrompt}(Q, \mathcal{C}_{\text{top}}) \tag{14}$$

In equation 14, where `FormatPrompt` ensures clear separation of the context from the question and incorporates instructions for generating grounded responses. The model used in this study is `gpt-4o-mini`. In the end, LLM integrates information from all selected chunks while maintaining semantic coherence and factual consistency. Since the scope of this study is to evaluate the impact of indexing, reranking, and similarity metrics on the RAG system, we have, for consistency, used only a single LLM.

J. EVALUATION

The performance of the proposed RAG pipeline is assessed through a comprehensive set of evaluation metrics, spanning the retrieval, reranking, and generation stages. This multilevel evaluation provides a detailed understanding of both the quality of information retrieval and the efficiency and effectiveness of the final generated answers.

1) RETRIEVAL AND RERANKING METRICS

The retrieval and reranking stages are evaluated using standard information retrieval metrics, capturing both accuracy and ranking quality:

- **Recall@k (R@k):** Recall@k measures the fraction of queries for which at least one relevant document appears in the top k retrieved results. Formally, for a set of queries Q and corresponding relevant documents R_q for query q, R@k is defined by equation 15

$$R@k = \frac{1}{|Q|} \sum_{q \in Q} \mathbf{1}\big(|\text{Top}_k(q) \cap R_q| \geq 1\big), \tag{15}$$

  $\text{Top}_k(q)$ denotes the set of top-k retrieved documents for query q, and $\mathbf{1}(\cdot)$ is the indicator function.
- **Mean Reciprocal Rank (MRR):** MRR evaluates the average rank position of the first relevant document in the retrieved list. It is computed using equation 16.

$$\text{MRR} = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{\text{rank}_q}, \tag{16}$$

  rank_q is the rank of the first relevant document for query q. Higher MRR indicates that relevant documents are ranked closer to the top.
- **Normalized Discounted Cumulative Gain (NDCG@k):** NDCG@k accounts for both relevance and position, assigning higher weight to relevant documents appearing earlier in the ranked list. It is calculated using equation 17.

$$\text{NDCG@k} = \frac{\text{DCG@k}}{\text{IDCG@k}}, \ \text{DCG@k} = \sum_{i=1}^{k} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)}, \tag{17}$$

  In equation 17 rel_i is the graded relevance of the document at rank i, and IDCG@k is the ideal DCG obtained by perfect ranking.
- **Coverage:** Coverage indicates the proportion of queries for which at least one relevant document is retrieved, providing a measure of the retrieval system's completeness across all queries.

2) COMPLETION METRICS

Once the top-ranked document chunks are identified and provided as context to the generative language model, the evaluation focuses on efficiency and cost-related metrics:

- **Latency:** The time required by the model to generate an answer for a given query, measured from prompt submission to response completion. Lower latency indicates better realtime performance.
- **Token Usage:** The total number of tokens consumed, including both prompt and completion tokens. Token usage reflects both computational efficiency and potential API costs when using commercial LLMs.
- **Cost:** Monetary cost per query, computed based on token usage and model pricing. This metric allows evaluation of economic efficiency in addition to technical performance.

IV. EVARAG EXPERIMENTAL DESIGN

To systematically evaluate the RAG pipeline, we designed experiments by combining multiple factors: dataset size, indexing method, similarity metric, retrieval strategy, and reranking approach. The parameters used are summarized in Algorithm 1.

All possible combinations of these parameters were generated to create the full set of experiments. The total number of experiments is calculated using the Cartesian product and are represented by 18

$$\begin{aligned} N_{\text{experiments}} &= |\mathcal{D}| \times |\mathcal{I}| \times |\mathcal{S}| \times |\mathcal{R}_{\text{ret}}| \times |\mathcal{R}_{\text{rerank}}| \\ &= 3 \times 3 \times 3 \times 3 \times 2 = 162. \end{aligned} \tag{18}$$

To illustrate the experimental design example, a subset of the generated RAG parameter combinations is shown in Table 2. Each row represents a unique experiment defined by the combination of dataset size, indexing method, similarity metric, retrieval strategy, and reranking approach. The experiments are numbered sequentially for clarity. Only a few examples are presented here to illustrate the structure; the complete set comprises 162 distinct experiments that cover all possible parameter combinations. These parameters were chosen because they reflect the most widely used and practically relevant design decisions in today's RAG

Algorithm 1 Generate All RAG Parameter Combinations

Require: Lists of parameters: DatasetList, IndexingList, SimilarityList, RetrievalList, RerankingList
Ensure: List of all possible RAG parameter combinations.

```
DatasetList ← {Small, Medium, Large}
IndexingList ← {HNSW, ScaNN, IVF}
SimilarityList ← {Cosine, Inner Product, L2}
RetrievalList ← {Fusion, Hierarchical, HyDE}
RerankingList ← {BGE, MiniLM, –}
Combinations ← empty list
for dataset in DatasetList do
  for indexing in IndexingList do
    for similarity in SimilarityList do
      for retrieval in RetrievalList do
        for reranking in RerankingList do
          Add (dataset, indexing similarity, retrieval,
          reranking) to Combinations.
        end for
      end for
    end for
  end for
end for
return Combinations
```

TABLE 2. Example combinations of experiments (Subset).

#	Dataset	Indexing	Similarity	Retrieval	Reranking
1	Small	HNSW	Cosine	Fusion	BGE
2	Small	HNSW	Cosine	Fusion	MiniLM
3	Small	HNSW	Cosine	HyDE	BGE
⋮	⋮	⋮	⋮	⋮	⋮
162	Large	IVF	L2	HyDE	MiniLM

pipelines. Dataset size, indexing structure, and similarity metric form the foundation of any retrieval system, while retrieval strategies such as Fusion, Hierarchical Search, and HyDE represent the prevailing paradigms found in the current literature. Likewise, BGE and MiniLM are lightweight rerankers commonly selected for scalable RAG solutions. Together, these parameters cover the full range of typical RAG configurations, ensuring that the experimental design is comprehensive, practical, and aligned with real-world system requirements.

Table 3 shows the implementation details for the EvaRAG pipeline, focusing on libraries, models, and APIs for reproducibility.

V. RESULTS

The experiments were implemented using Python and rely on several key libraries langchain, langchain-openai, langchain_community, pymilvus, and langchain-milvus. Milvus Lite was used for vector storage and retrieval. All experiments were performed on a MacBook Pro (Late 2019) with Intel Core i9 8-Core CPU (2.3 GHz), 32 GB DDR4 RAM (2667 MHz), macOS 15.6.1. Let's discuss the empirical experiments in detail.

TABLE 3. Key implementation details of EvaRAG pipeline focusing on libraries, models, and APIs for reproducibility.

Category	Implementation Details
Programming Language	Python
Libraries	LangChain, LangChain-OpenAI, LangChain-Community, Python-Dotenv, Sentence-Transformers, Deepeval, Datasets, NumPy (<2), Milvus
Vector Database	Milvus (HNSW, IVF, SCANN indices)
API	OpenAI (Embeddings and LLM)

A. RETRIEVAL RESULTS

Table 4 reports the retrieval performance of HNSW using the Cosine metric across retrievers, rerankers, and dataset sizes. Overall, the Fusion retriever achieves the best results, with BGE reranking yielding R1 = 0.733, R3 = 0.870, MRR = 0.807, and $nDCG@10 = 0.839$ on the Medium dataset, and slightly lower but comparable performance on Large datasets. The Hierarchical retriever performs poorly on Large datasets (R1 = 0.448, MRR = 0.499), but improves substantially on Small datasets (R1 = 0.655, MRR = 0.727). The HyDe retriever provides a middle ground, e.g., the Medium dataset with BGE reranker achieves R1 = 0.612, R3 = 0.786, and $nDCG@10 = 0.751$. These results indicate that Fusion + BGE is the most effective configuration, Hierarchical benefits from smaller corpora, and HyDe offers balanced performance.

Table 5 presents the retrieval performance of the HNSW index using the Inner Product (IP) metric across different retrievers, rerankers, and dataset sizes. The results show that the Fusion retriever consistently outperforms the Hierarchical and HyDe retrievers across all dataset sizes, with R@1 (R1) ranging from 0.720 to 0.750 for Fusion, compared to 0.454 to 0.569 for Hierarchical and 0.578 to 0.626 for HyDe. Similarly, R@3 (R3) and R@10 (R10) follow the same trend, with Fusion achieving up to 0.887 and 0.942, respectively. The choice of reranker also impacts performance: BGE and minilm show nearly identical results for Fusion, while minilm slightly improves R@1–R@10 for Hierarchical and HyDe. MRR values mirror these trends, with Fusion reaching a maximum of 0.823, compared to 0.627 for Hierarchical and 0.719 for HyDe. The nDCG metrics (nDCG1, nDCG3, nDCG10) indicate that Fusion retrieves more relevant items in top positions, achieving an nDCG10 of up to 0.852, while Hierarchical and HyDe lag behind at 0.650 and 0.758, respectively. Overall, the table demonstrates that both retriever choice and reranker selection have a substantial effect on retrieval quality, and that the Fusion retriever with either BGE or minilm reranker provides the most effective combination for HNSW with the IP metric.

Table 6 presents the retrieval performance of the HNSW index using the Euclidean (L2) distance across different retrievers, rerankers, and dataset sizes. The Fusion retriever

TABLE 4. Retrieval performance of HNSW with Cosine Metrics across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
HNSW	Cosine	Fusion	BGE	Large	0.708	0.840	0.911	0.782	0.708	0.787	0.814
HNSW	Cosine	Fusion	BGE	Medium	0.733	0.870	0.935	0.807	0.733	0.815	0.839
HNSW	Cosine	Fusion	BGE	Small	0.723	0.858	0.917	0.796	0.723	0.804	0.826
HNSW	Cosine	Fusion	minilm	Large	0.708	0.840	0.911	0.782	0.708	0.787	0.814
HNSW	Cosine	Fusion	minilm	Medium	0.733	0.870	0.935	0.807	0.733	0.815	0.839
HNSW	Cosine	Fusion	minilm	Small	0.723	0.858	0.917	0.796	0.723	0.804	0.826
HNSW	Cosine	Hierarchical	BGE	Large	0.448	0.541	0.589	0.499	0.448	0.503	0.521
HNSW	Cosine	Hierarchical	BGE	Medium	0.488	0.591	0.640	0.543	0.488	0.549	0.567
HNSW	Cosine	Hierarchical	BGE	Small	0.655	0.795	0.849	0.727	0.655	0.738	0.757
HNSW	Cosine	Hierarchical	minilm	Large	0.448	0.541	0.589	0.499	0.448	0.503	0.521
HNSW	Cosine	Hierarchical	minilm	Medium	0.488	0.591	0.640	0.543	0.488	0.549	0.567
HNSW	Cosine	Hierarchical	minilm	Small	0.655	0.795	0.849	0.727	0.655	0.738	0.757
HNSW	Cosine	HyDe	BGE	Large	0.580	0.757	0.854	0.679	0.580	0.685	0.722
HNSW	Cosine	HyDe	BGE	Medium	0.612	0.786	0.883	0.708	0.612	0.715	0.751
HNSW	Cosine	HyDe	BGE	Small	0.622	0.795	0.873	0.715	0.622	0.725	0.754
HNSW	Cosine	HyDe	minilm	Large	0.580	0.757	0.854	0.679	0.580	0.685	0.722
HNSW	Cosine	HyDe	minilm	Medium	0.612	0.786	0.883	0.708	0.612	0.715	0.751
HNSW	Cosine	HyDe	minilm	Small	0.622	0.795	0.873	0.715	0.622	0.725	0.754

TABLE 5. Retrieval performance of HNSW with Inner Product (IP) across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
HNSW	IP	Fusion	BGE	Large	0.720	0.857	0.925	0.793	0.720	0.801	0.825
HNSW	IP	Fusion	BGE	Medium	0.748	0.881	0.938	0.819	0.748	0.827	0.848
HNSW	IP	Fusion	BGE	Small	0.750	0.887	0.942	0.823	0.750	0.832	0.852
HNSW	IP	Fusion	minilm	Large	0.720	0.857	0.925	0.793	0.720	0.801	0.825
HNSW	IP	Fusion	minilm	Medium	0.748	0.881	0.938	0.819	0.748	0.827	0.848
HNSW	IP	Fusion	minilm	Small	0.750	0.887	0.942	0.823	0.750	0.832	0.852
HNSW	IP	Hierarchical	BGE	Large	0.454	0.547	0.594	0.505	0.454	0.509	0.527
HNSW	IP	Hierarchical	BGE	Medium	0.500	0.603	0.649	0.555	0.500	0.561	0.578
HNSW	IP	Hierarchical	BGE	Small	0.569	0.680	0.723	0.627	0.569	0.635	0.650
HNSW	IP	Hierarchical	minilm	Large	0.454	0.547	0.594	0.505	0.454	0.509	0.527
HNSW	IP	Hierarchical	minilm	Medium	0.500	0.603	0.649	0.555	0.500	0.561	0.578
HNSW	IP	Hierarchical	minilm	Small	0.569	0.680	0.723	0.627	0.569	0.635	0.650
HNSW	IP	HyDe	BGE	Large	0.578	0.754	0.851	0.676	0.578	0.683	0.718
HNSW	IP	HyDe	BGE	Medium	0.610	0.786	0.882	0.707	0.610	0.714	0.750
HNSW	IP	HyDe	BGE	Small	0.626	0.800	0.879	0.719	0.626	0.729	0.758
HNSW	IP	HyDe	minilm	Large	0.578	0.754	0.851	0.676	0.578	0.683	0.718
HNSW	IP	HyDe	minilm	Medium	0.610	0.786	0.882	0.707	0.610	0.714	0.750
HNSW	IP	HyDe	minilm	Small	0.626	0.800	0.879	0.719	0.626	0.729	0.758

performs poorly with L2, achieving very low R1 (0.030–0.046) and R10 (0.271–0.373), indicating that top-ranked results rarely contain the correct item. Hierarchical retrievers improve substantially with R1 ranging 0.481–0.572 and R10 0.631–0.731, while HyDe achieves the highest performance among all retrievers (R1 0.580–0.625, R10 0.856–0.888). MRR and nDCG metrics follow similar trends, showing that Fusion fails to rank relevant items effectively with L2, whereas HyDe maintains relatively high reciprocal rank and relevance-aware ranking (nDCG10 up to 0.763). Overall, these results highlight that the choice of distance metric has a drastic impact on retrieval quality, with L2 being unsuitable for Fusion retrievers but effective with HyDe and Hierarchical configurations.

Table 7 reports the retrieval performance of the IVF index using the Cosine metric across various retrievers, rerankers, and dataset sizes. Fusion retrievers achieve the highest performance, with R1 ranging from 0.683 to 0.736 and R10 ranging from 0.870 to 0.927, indicating that relevant items are frequently ranked within the top results. HyDe retrievers provide moderate performance (R1 0.565–0.618, R10 0.828–0.870), while Hierarchical retrievers perform the lowest (R1 0.409–0.540, R10 0.535–0.688). MRR and nDCG metrics reflect similar trends, indicating that Fusion consistently ranks correct results higher and more accurately account for relevance in the top positions. Overall, the choice of retriever significantly impacts retrieval quality, with Fusion outperforming both Hierarchical and HyDe across all dataset sizes.

Table 8 presents the retrieval performance of the IVF index with the Inner Product (IP) metric across different retrievers, rerankers, and dataset sizes. Fusion retrievers achieve the highest performance, with R1 ranging from 0.684 to 0.752 and R10 from 0.910 to 0.937, indicating

TABLE 6. Retrieval performance of HNSW with Euclidean Distance (L2) across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
HNSW	L2	Fusion	BGE	Large	0.046	0.110	0.373	0.115	0.046	0.082	0.175
HNSW	L2	Fusion	BGE	Medium	0.036	0.093	0.317	0.094	0.036	0.067	0.145
HNSW	L2	Fusion	BGE	Small	0.030	0.068	0.271	0.077	0.030	0.051	0.121
HNSW	L2	Fusion	minilm	Large	0.046	0.110	0.373	0.115	0.046	0.082	0.175
HNSW	L2	Fusion	minilm	Medium	0.036	0.093	0.317	0.094	0.036	0.067	0.145
HNSW	L2	Fusion	minilm	Small	0.030	0.068	0.271	0.077	0.030	0.051	0.121
HNSW	L2	Hierarchical	BGE	Large	0.481	0.581	0.631	0.535	0.481	0.540	0.559
HNSW	L2	Hierarchical	BGE	Medium	0.497	0.604	0.653	0.554	0.497	0.560	0.578
HNSW	L2	Hierarchical	BGE	Small	0.572	0.686	0.731	0.631	0.572	0.639	0.655
HNSW	L2	Hierarchical	minilm	Large	0.481	0.581	0.631	0.535	0.481	0.540	0.559
HNSW	L2	Hierarchical	minilm	Medium	0.497	0.604	0.653	0.554	0.497	0.560	0.578
HNSW	L2	Hierarchical	minilm	Small	0.572	0.686	0.731	0.631	0.572	0.639	0.655
HNSW	L2	HyDe	BGE	Large	0.580	0.758	0.856	0.679	0.580	0.686	0.722
HNSW	L2	HyDe	BGE	Medium	0.611	0.787	0.879	0.707	0.611	0.715	0.749
HNSW	L2	HyDe	BGE	Small	0.625	0.804	0.888	0.722	0.625	0.732	0.763
HNSW	L2	HyDe	minilm	Large	0.580	0.758	0.856	0.679	0.580	0.686	0.722
HNSW	L2	HyDe	minilm	Medium	0.611	0.787	0.879	0.707	0.611	0.715	0.749
HNSW	L2	HyDe	minilm	Small	0.625	0.804	0.888	0.722	0.625	0.732	0.763

TABLE 7. Retrieval performance of IVF with Cosine Metrics across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
IVF	Cosine	Fusion	BGE	Large	0.683	0.806	0.870	0.751	0.683	0.756	0.780
IVF	Cosine	Fusion	BGE	Medium	0.706	0.833	0.891	0.774	0.706	0.782	0.803
IVF	Cosine	Fusion	BGE	Small	0.736	0.870	0.927	0.809	0.736	0.817	0.838
IVF	Cosine	Fusion	minilm	Large	0.683	0.806	0.870	0.751	0.683	0.756	0.780
IVF	Cosine	Fusion	minilm	Medium	0.706	0.833	0.891	0.774	0.706	0.782	0.803
IVF	Cosine	Fusion	minilm	Small	0.736	0.870	0.927	0.809	0.736	0.817	0.838
IVF	Cosine	Hierarchical	BGE	Large	0.409	0.494	0.535	0.456	0.409	0.460	0.475
IVF	Cosine	Hierarchical	BGE	Medium	0.476	0.578	0.620	0.530	0.476	0.537	0.552
IVF	Cosine	Hierarchical	BGE	Small	0.540	0.643	0.688	0.595	0.540	0.602	0.617
IVF	Cosine	Hierarchical	minilm	Large	0.409	0.494	0.535	0.456	0.409	0.460	0.475
IVF	Cosine	Hierarchical	minilm	Medium	0.476	0.578	0.620	0.530	0.476	0.537	0.552
IVF	Cosine	Hierarchical	minilm	Small	0.540	0.643	0.688	0.595	0.540	0.602	0.617
IVF	Cosine	HyDe	BGE	Large	0.565	0.734	0.828	0.659	0.565	0.665	0.700
IVF	Cosine	HyDe	BGE	Medium	0.601	0.771	0.859	0.694	0.601	0.701	0.734
IVF	Cosine	HyDe	BGE	Small	0.618	0.794	0.870	0.711	0.618	0.722	0.750
IVF	Cosine	HyDe	minilm	Large	0.565	0.734	0.828	0.659	0.565	0.665	0.700
IVF	Cosine	HyDe	minilm	Medium	0.601	0.771	0.859	0.694	0.601	0.701	0.734
IVF	Cosine	HyDe	minilm	Small	0.618	0.794	0.870	0.711	0.618	0.722	0.750

that relevant items are consistently retrieved within the top ranks. HyDe retrievers show moderate performance (R1 0.566–0.625, R10 0.832–0.880), while Hierarchical retrievers generally perform lower (R1 0.434–0.643, R10 0.567–0.838). MRR and nDCG metrics align with this trend, demonstrating that Fusion retrievers rank the correct results higher and more accurately account for relevance in top positions. Overall, the results highlight that Fusion-based retrieval with IP outperforms both Hierarchical and HyDe across all dataset sizes.

Table 9 presents the retrieval performance of the IVF index with Euclidean Distance (L2) across different retrievers, rerankers, and dataset sizes. Fusion retrievers exhibit very low performance, with R1 ranging from 0.023 to 0.037 and R10 ranging from 0.271 to 0.370, indicating poor retrieval for top-ranked items. Hierarchical retrievers improve considerably (R1 0.431–0.553, R10 0.556–0.698), while HyDe retrievers achieve the highest L2-based performance (R1 0.574–0.610, R10 0.822–0.857). MRR and nDCG metrics follow the same trend, reflecting that HyDe retrievers rank relevant results higher and more accurately consider relevance at top positions. Overall, L2 distance performs worse for Fusion but remains effective for HyDe and Hierarchical retrievers across all dataset sizes.

Table 10 reports the retrieval performance of SCANN with Cosine similarity across retrievers, rerankers, and dataset sizes. Fusion retrievers consistently perform best, achieving R1 values between 0.690 and 0.733 and R10 values of up to 0.923, indicating highly accurate top-ranked retrieval. HyDe retrievers follow with moderate performance (R1 0.569–0.622, R10 0.830–0.877), while Hierarchical retrievers remain the weakest (R1 0.440–0.589, R10 0.578–0.759). MRR and nDCG values show the same ranking pattern, confirming that SCANN with Cosine is most effective

TABLE 8. Retrieval performance of IVF with Inner Product (IP) across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
IVF	IP	Fusion	BGE	Large	0.684	0.823	0.910	0.761	0.684	0.766	0.797
IVF	IP	Fusion	BGE	Medium	0.727	0.851	0.923	0.795	0.727	0.800	0.826
IVF	IP	Fusion	BGE	Small	0.752	0.886	0.937	0.822	0.752	0.832	0.850
IVF	IP	Fusion	minilm	Large	0.684	0.823	0.910	0.761	0.684	0.766	0.797
IVF	IP	Fusion	minilm	Medium	0.727	0.851	0.923	0.795	0.727	0.800	0.826
IVF	IP	Fusion	minilm	Small	0.752	0.886	0.937	0.822	0.752	0.832	0.850
IVF	IP	Hierarchical	BGE	Large	0.434	0.524	0.567	0.483	0.434	0.487	0.503
IVF	IP	Hierarchical	BGE	Medium	0.483	0.583	0.632	0.537	0.483	0.542	0.560
IVF	IP	Hierarchical	BGE	Small	0.643	0.781	0.838	0.715	0.643	0.725	0.745
IVF	IP	Hierarchical	minilm	Large	0.434	0.524	0.567	0.483	0.434	0.487	0.503
IVF	IP	Hierarchical	minilm	Medium	0.483	0.583	0.632	0.537	0.483	0.542	0.560
IVF	IP	Hierarchical	minilm	Small	0.643	0.781	0.838	0.715	0.643	0.725	0.745
IVF	IP	HyDe	BGE	Large	0.566	0.735	0.832	0.661	0.566	0.666	0.703
IVF	IP	HyDe	BGE	Medium	0.601	0.768	0.858	0.693	0.601	0.700	0.733
IVF	IP	HyDe	BGE	Small	0.625	0.803	0.880	0.720	0.625	0.731	0.759
IVF	IP	HyDe	minilm	Large	0.566	0.735	0.832	0.661	0.566	0.666	0.703
IVF	IP	HyDe	minilm	Medium	0.601	0.768	0.858	0.693	0.601	0.700	0.733
IVF	IP	HyDe	minilm	Small	0.625	0.803	0.880	0.720	0.625	0.731	0.759

TABLE 9. Retrieval performance of IVF with Euclidean Distance (L2) across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
IVF	l2	Fusion	BGE	Large	0.037	0.108	0.370	0.110	0.037	0.078	0.169
IVF	l2	Fusion	BGE	Medium	0.031	0.093	0.323	0.093	0.031	0.066	0.145
IVF	l2	Fusion	BGE	Small	0.023	0.066	0.271	0.073	0.023	0.047	0.118
IVF	l2	Fusion	minilm	Large	0.037	0.108	0.370	0.110	0.037	0.078	0.169
IVF	l2	Fusion	minilm	Medium	0.031	0.093	0.323	0.093	0.031	0.066	0.145
IVF	l2	Fusion	minilm	Small	0.023	0.066	0.271	0.073	0.023	0.047	0.118
IVF	l2	Hierarchical	BGE	Large	0.431	0.514	0.556	0.476	0.431	0.480	0.495
IVF	l2	Hierarchical	BGE	Medium	0.488	0.588	0.638	0.543	0.488	0.548	0.566
IVF	l2	Hierarchical	BGE	Small	0.553	0.650	0.698	0.605	0.553	0.611	0.628
IVF	l2	Hierarchical	minilm	Large	0.431	0.514	0.556	0.476	0.431	0.480	0.495
IVF	l2	Hierarchical	minilm	Medium	0.488	0.588	0.638	0.543	0.488	0.548	0.566
IVF	l2	Hierarchical	minilm	Small	0.553	0.650	0.698	0.605	0.553	0.611	0.628
IVF	l2	HyDe	BGE	Large	0.574	0.733	0.822	0.663	0.574	0.669	0.702
IVF	l2	HyDe	BGE	Medium	0.593	0.762	0.849	0.686	0.593	0.693	0.726
IVF	l2	HyDe	BGE	Small	0.610	0.781	0.857	0.701	0.610	0.711	0.740
IVF	l2	HyDe	minilm	Large	0.574	0.733	0.822	0.663	0.574	0.669	0.702
IVF	l2	HyDe	minilm	Medium	0.593	0.762	0.849	0.686	0.593	0.693	0.726
IVF	l2	HyDe	minilm	Small	0.610	0.781	0.857	0.701	0.610	0.711	0.740

with Fusion retrievers, especially on smaller datasets, where performance peaks.

Table 11 reports the retrieval performance of SCANN with Inner Product (IP) across retrievers, rerankers, and dataset sizes. Fusion retrievers consistently achieve the highest performance, with R1 ranging from 0.696 to 0.752 and R10 reaching 0.911 to 0.937, demonstrating strong top-ranked retrieval accuracy. HyDe retrievers perform moderately well (R1 0.572–0.627, R10 0.833–0.884), while Hierarchical retrievers lag, especially on large datasets (R1 as low as 0.422, R10 0.559). MRR and nDCG values follow the same trend, confirming that SCANN with IP combined with Fusion retrievers provides the most effective retrieval across dataset sizes.

Table 12 presents the retrieval performance of SCANN with Euclidean Distance (L2). Fusion retrievers perform poorly, with R1 as low as 0.022–0.038 and R10 below 0.36, indicating weak top-ranked retrieval quality. Hierarchical retrievers perform moderately, with improvements observed in smaller datasets (R1: 0.398–0.558, R10: 0.506–0.704). HyDe retrievers consistently achieve the best results for L2, with R1 between 0.573 to 0.613 and R10 reaching up to 0.865, clearly outperforming other retrievers under this metric. MRR and nDCG trends mirror these results, confirming that SCANN with L2 is most effective when paired with HyDe retrievers.

B. RERANKING RESULTS

The reranking results of HNSW with Cosine Similarity are summarized in Table 13. Overall, performance varies across retrievers, rerankers, and dataset sizes, with clear trends indicating that larger datasets generally improve retrieval effectiveness. Among retrievers, Fusion combined

TABLE 10. Retrieval performance of SCANN with Cosine across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
SCANN	Cosine	Fusion	BGE	Large	0.690	0.822	0.885	0.762	0.690	0.768	0.792
SCANN	Cosine	Fusion	BGE	Medium	0.706	0.834	0.895	0.776	0.706	0.783	0.805
SCANN	Cosine	Fusion	BGE	Small	0.733	0.867	0.923	0.806	0.733	0.814	0.835
SCANN	Cosine	Fusion	minilm	Large	0.690	0.822	0.885	0.762	0.690	0.768	0.792
SCANN	Cosine	Fusion	minilm	Medium	0.706	0.834	0.895	0.776	0.706	0.783	0.805
SCANN	Cosine	Fusion	minilm	Small	0.733	0.867	0.923	0.806	0.733	0.814	0.835
SCANN	Cosine	Hierarchical	BGE	Large	0.440	0.531	0.578	0.490	0.440	0.494	0.511
SCANN	Cosine	Hierarchical	BGE	Medium	0.496	0.595	0.642	0.549	0.496	0.554	0.572
SCANN	Cosine	Hierarchical	BGE	Small	0.589	0.713	0.759	0.653	0.589	0.662	0.679
SCANN	Cosine	Hierarchical	minilm	Large	0.440	0.531	0.578	0.490	0.440	0.494	0.511
SCANN	Cosine	Hierarchical	minilm	Medium	0.496	0.595	0.642	0.549	0.496	0.554	0.572
SCANN	Cosine	Hierarchical	minilm	Small	0.589	0.713	0.759	0.653	0.589	0.662	0.679
SCANN	Cosine	HyDe	BGE	Large	0.569	0.735	0.830	0.662	0.569	0.668	0.703
SCANN	Cosine	HyDe	BGE	Medium	0.599	0.766	0.852	0.691	0.599	0.698	0.731
SCANN	Cosine	HyDe	BGE	Small	0.622	0.798	0.877	0.716	0.622	0.726	0.756
SCANN	Cosine	HyDe	minilm	Large	0.569	0.735	0.830	0.662	0.569	0.668	0.703
SCANN	Cosine	HyDe	minilm	Medium	0.599	0.766	0.852	0.691	0.599	0.698	0.731
SCANN	Cosine	HyDe	minilm	Small	0.622	0.798	0.877	0.716	0.622	0.726	0.756

TABLE 11. Retrieval performance of SCANN with Inner Product (IP) across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
SCANN	IP	Fusion	BGE	Large	0.696	0.830	0.911	0.770	0.696	0.775	0.804
SCANN	IP	Fusion	BGE	Medium	0.726	0.849	0.925	0.795	0.726	0.799	0.827
SCANN	IP	Fusion	BGE	Small	0.752	0.886	0.937	0.822	0.752	0.832	0.850
SCANN	IP	Fusion	minilm	Large	0.696	0.830	0.911	0.770	0.696	0.775	0.804
SCANN	IP	Fusion	minilm	Medium	0.726	0.849	0.925	0.795	0.726	0.799	0.827
SCANN	IP	Fusion	minilm	Small	0.752	0.886	0.937	0.822	0.752	0.832	0.850
SCANN	IP	Hierarchical	BGE	Large	0.422	0.513	0.559	0.471	0.422	0.476	0.493
SCANN	IP	Hierarchical	BGE	Medium	0.488	0.592	0.643	0.544	0.488	0.549	0.568
SCANN	IP	Hierarchical	BGE	Small	0.638	0.778	0.831	0.711	0.638	0.721	0.740
SCANN	IP	Hierarchical	minilm	Large	0.422	0.513	0.559	0.471	0.422	0.476	0.493
SCANN	IP	Hierarchical	minilm	Medium	0.488	0.592	0.643	0.544	0.488	0.549	0.568
SCANN	IP	Hierarchical	minilm	Small	0.638	0.778	0.831	0.711	0.638	0.721	0.740
SCANN	IP	HyDe	BGE	Large	0.572	0.739	0.833	0.665	0.572	0.671	0.706
SCANN	IP	HyDe	BGE	Medium	0.600	0.768	0.851	0.692	0.600	0.700	0.731
SCANN	IP	HyDe	BGE	Small	0.627	0.803	0.884	0.721	0.627	0.731	0.761
SCANN	IP	HyDe	minilm	Large	0.572	0.739	0.833	0.665	0.572	0.671	0.706
SCANN	IP	HyDe	minilm	Medium	0.600	0.768	0.851	0.692	0.600	0.700	0.731
SCANN	IP	HyDe	minilm	Small	0.627	0.803	0.884	0.721	0.627	0.731	0.761

with minilm reranker consistently yields the strongest performance, achieving R1 scores above 0.80 for all dataset sizes and reaching a peak of 0.82 on the medium dataset, alongside high nDCG values (nDCG1 = 0.872). This demonstrates that minilm reranking is highly effective when paired with Fusion retrieval. In contrast, Hierarchical retrieval shows relatively weaker performance, with R1 scores ranging from 0.33 to 0.75, depending on the dataset size; however, its performance improves markedly on the smaller dataset. HyDe retrievers perform moderately, with minilm reranking again outperforming BGE reranking, achieving R1 scores up to 0.77. Across all configurations, MRR trends closely mirror R1, confirming that systems retrieving correct items earlier also deliver higher overall ranking quality. The consistently strong nDCG3 and nDCG10 scores for Fusion + minilm configurations indicate that not only are correct results found early, but other relevant items are also ranked appropriately. These findings highlight that pairing HNSW with cosine similarity, Fusion retrieval, and minilm reranking is the most effective configuration for achieving high top-K recall and ranking quality across dataset scales.

Table 14 presents the reranking performance of HNSW with Inner Product (IP). Overall, performance trends are highly consistent with those observed for cosine similarity, though IP yields slightly higher scores in several configurations. Fusion retrieval with minilm reranking again stands out as the strongest configuration, achieving the highest R1 (0.825) and MRR (0.872) on the small dataset, with similarly strong results across medium and large datasets. This indicates that IP based HNSW retrieval benefits from the dense representation power of Fusion combined with minilm's fine-grained reranking. Fusion + BGE also performs well, with R1 values around 0.58–0.58 and steadily improving nDCG scores as dataset size grows. Hierarchical

TABLE 12. Retrieval performance of SCANN with Euclidean Distance (L2) across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
SCANN	l2	Fusion	BGE	Large	0.038	0.104	0.354	0.107	0.038	0.076	0.163
SCANN	l2	Fusion	BGE	Medium	0.028	0.094	0.323	0.092	0.028	0.066	0.144
SCANN	l2	Fusion	BGE	Small	0.022	0.065	0.267	0.071	0.022	0.046	0.115
SCANN	l2	Fusion	minilm	Large	0.038	0.104	0.354	0.107	0.038	0.076	0.163
SCANN	l2	Fusion	minilm	Medium	0.028	0.094	0.323	0.092	0.028	0.066	0.144
SCANN	l2	Fusion	minilm	Small	0.022	0.065	0.267	0.071	0.022	0.046	0.115
SCANN	l2	Hierarchical	BGE	Large	0.398	0.474	0.506	0.438	0.398	0.443	0.455
SCANN	l2	Hierarchical	BGE	Medium	0.434	0.522	0.558	0.480	0.434	0.486	0.499
SCANN	l2	Hierarchical	BGE	Small	0.558	0.655	0.704	0.610	0.558	0.615	0.633
SCANN	l2	Hierarchical	minilm	Large	0.398	0.474	0.506	0.438	0.398	0.443	0.455
SCANN	l2	Hierarchical	minilm	Medium	0.434	0.522	0.558	0.480	0.434	0.486	0.499
SCANN	l2	Hierarchical	minilm	Small	0.558	0.655	0.704	0.610	0.558	0.615	0.633
SCANN	l2	HyDe	BGE	Large	0.573	0.738	0.829	0.665	0.573	0.670	0.705
SCANN	l2	HyDe	BGE	Medium	0.592	0.758	0.845	0.683	0.592	0.690	0.723
SCANN	l2	HyDe	BGE	Small	0.613	0.790	0.865	0.707	0.613	0.718	0.746
SCANN	l2	HyDe	minilm	Large	0.573	0.738	0.829	0.665	0.573	0.670	0.705
SCANN	l2	HyDe	minilm	Medium	0.592	0.758	0.845	0.683	0.592	0.690	0.723
SCANN	l2	HyDe	minilm	Small	0.613	0.790	0.865	0.707	0.613	0.718	0.746

TABLE 13. Reranking performance of HNSW with Cosine Similarity across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
HNSW	Cosine	Fusion	BGE	Large	0.548	0.786	0.911	0.677	0.548	0.687	0.734
HNSW	Cosine	Fusion	BGE	Medium	0.553	0.796	0.935	0.688	0.553	0.695	0.749
HNSW	Cosine	Fusion	BGE	Small	0.538	0.778	0.917	0.669	0.538	0.677	0.730
HNSW	Cosine	Fusion	minilm	Large	0.798	0.879	0.911	0.841	0.798	0.846	0.858
HNSW	Cosine	Fusion	minilm	Medium	0.820	0.907	0.935	0.866	0.820	0.872	0.883
HNSW	Cosine	Fusion	minilm	Small	0.804	0.890	0.917	0.849	0.804	0.856	0.866
HNSW	Cosine	Hierarchical	BGE	Large	0.333	0.479	0.589	0.418	0.333	0.418	0.460
HNSW	Cosine	Hierarchical	BGE	Medium	0.346	0.515	0.640	0.443	0.346	0.444	0.491
HNSW	Cosine	Hierarchical	BGE	Small	0.464	0.683	0.849	0.591	0.464	0.591	0.654
HNSW	Cosine	Hierarchical	minilm	Large	0.507	0.570	0.589	0.539	0.507	0.544	0.551
HNSW	Cosine	Hierarchical	minilm	Medium	0.554	0.621	0.640	0.588	0.554	0.594	0.601
HNSW	Cosine	Hierarchical	minilm	Small	0.754	0.827	0.849	0.792	0.754	0.798	0.806
HNSW	Cosine	HyDe	BGE	Large	0.488	0.695	0.854	0.610	0.488	0.609	0.669
HNSW	Cosine	HyDe	BGE	Medium	0.498	0.716	0.883	0.625	0.498	0.625	0.688
HNSW	Cosine	HyDe	BGE	Small	0.489	0.698	0.873	0.614	0.489	0.611	0.677
HNSW	Cosine	HyDe	minilm	Large	0.746	0.824	0.854	0.787	0.746	0.793	0.804
HNSW	Cosine	HyDe	minilm	Medium	0.771	0.852	0.883	0.814	0.771	0.819	0.831
HNSW	Cosine	HyDe	minilm	Small	0.769	0.845	0.873	0.810	0.769	0.815	0.826

retrieval remains weaker overall but improves steadily with dataset size, especially when paired with minilm reranking (R1 = 0.638 on small dataset). HyDe retrievers perform competitively in combination with minilm, achieving R1 up to 0.775 and MRR above 0.81. The close alignment between R1, MRR, and nDCG metrics confirms that systems not only retrieve the correct items early but also effectively rank multiple relevant items. These results suggest that HNSW with IP is a strong alternative to cosine similarity, with slightly better retrieval quality in many cases, particularly for Fusion + minilm configurations.

Table 15 reports the reranking performance of HNSW with Euclidean Distance (L2). Compared to cosine and inner-product similarity, L2 generally produces lower R1 and MRR scores for Fusion based retrieval, with Fusion + minilm achieving only R1 = 0.335 on the large dataset. However, Hierarchical retrieval benefits more under L2, especially with minilm reranking, where performance improves significantly as dataset size decreases, reaching R1 = 0.648 and MRR = 0.680 on the small dataset. HyDe retrievers continue to perform competitively, particularly when combined with minilm reranking, consistently achieving R1 above 0.74 and strong nDCG scores across all dataset sizes, with the best result on the small dataset (R1 = 0.783, nDCG1 = 0.830). The results indicate that while L2 distance is less effective for dense retrievers like Fusion, it still performs well with Hierarchical and HyDe retrieval, especially when reranking with minilm, which consistently yields the highest ranking quality (MRR and nDCG values).

Table 16 presents the reranking performance of IVF with Cosine similarity. Fusion-based retrieval with minilm reranking consistently achieves the strongest results across all dataset sizes, with R1 increasing from 0.760 (large) to 0.818 (small) and MRR reaching 0.862, highlighting

TABLE 14. Reranking performance of HNSW with Inner Product (IP) across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
HNSW	IP	Fusion	BGE	Large	0.580	0.809	0.925	0.705	0.580	0.714	0.759
HNSW	IP	Fusion	BGE	Medium	0.582	0.812	0.938	0.708	0.582	0.715	0.764
HNSW	IP	Fusion	BGE	Small	0.576	0.806	0.942	0.703	0.576	0.709	0.761
HNSW	IP	Fusion	minilm	Large	0.807	0.893	0.925	0.852	0.807	0.858	0.870
HNSW	IP	Fusion	minilm	Medium	0.824	0.910	0.938	0.870	0.824	0.876	0.887
HNSW	IP	Fusion	minilm	Small	0.825	0.913	0.942	0.872	0.825	0.878	0.889
HNSW	IP	Hierarchical	BGE	Large	0.333	0.481	0.594	0.419	0.333	0.419	0.461
HNSW	IP	Hierarchical	BGE	Medium	0.361	0.524	0.649	0.456	0.361	0.456	0.503
HNSW	IP	Hierarchical	BGE	Small	0.401	0.578	0.723	0.506	0.401	0.504	0.559
HNSW	IP	Hierarchical	minilm	Large	0.516	0.576	0.594	0.547	0.516	0.552	0.558
HNSW	IP	Hierarchical	minilm	Medium	0.568	0.633	0.649	0.601	0.568	0.607	0.613
HNSW	IP	Hierarchical	minilm	Small	0.638	0.701	0.723	0.672	0.638	0.676	0.685
HNSW	IP	HyDe	BGE	Large	0.483	0.692	0.851	0.606	0.483	0.606	0.666
HNSW	IP	HyDe	BGE	Medium	0.498	0.715	0.882	0.624	0.498	0.624	0.687
HNSW	IP	HyDe	BGE	Small	0.493	0.706	0.879	0.620	0.493	0.618	0.682
HNSW	IP	HyDe	minilm	Large	0.742	0.821	0.851	0.783	0.742	0.789	0.800
HNSW	IP	HyDe	minilm	Medium	0.769	0.851	0.882	0.813	0.769	0.818	0.830
HNSW	IP	HyDe	minilm	Small	0.775	0.853	0.879	0.816	0.775	0.822	0.832

TABLE 15. Reranking performance of HNSW with Euclidean Distance (L2) across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
HNSW	l2	Fusion	BGE	Large	0.252	0.334	0.373	0.296	0.252	0.300	0.315
HNSW	l2	Fusion	BGE	Medium	0.218	0.283	0.317	0.253	0.218	0.255	0.268
HNSW	l2	Fusion	BGE	Small	0.177	0.236	0.271	0.210	0.177	0.212	0.225
HNSW	l2	Fusion	minilm	Large	0.335	0.363	0.373	0.350	0.335	0.352	0.356
HNSW	l2	Fusion	minilm	Medium	0.292	0.308	0.317	0.301	0.292	0.301	0.305
HNSW	l2	Fusion	minilm	Small	0.243	0.264	0.271	0.254	0.243	0.256	0.258
HNSW	l2	Hierarchical	BGE	Large	0.358	0.521	0.631	0.450	0.358	0.453	0.494
HNSW	l2	Hierarchical	BGE	Medium	0.363	0.528	0.653	0.458	0.363	0.458	0.505
HNSW	l2	Hierarchical	BGE	Small	0.398	0.592	0.731	0.509	0.398	0.511	0.563
HNSW	l2	Hierarchical	minilm	Large	0.549	0.613	0.631	0.581	0.549	0.587	0.594
HNSW	l2	Hierarchical	minilm	Medium	0.572	0.634	0.653	0.604	0.572	0.609	0.616
HNSW	l2	Hierarchical	minilm	Small	0.648	0.710	0.731	0.680	0.648	0.685	0.693
HNSW	l2	HyDe	BGE	Large	0.491	0.698	0.856	0.613	0.491	0.613	0.672
HNSW	l2	HyDe	BGE	Medium	0.499	0.712	0.879	0.624	0.499	0.623	0.686
HNSW	l2	HyDe	BGE	Small	0.500	0.714	0.888	0.625	0.500	0.624	0.689
HNSW	l2	HyDe	minilm	Large	0.748	0.826	0.856	0.789	0.748	0.795	0.806
HNSW	l2	HyDe	minilm	Medium	0.768	0.850	0.879	0.811	0.768	0.817	0.828
HNSW	l2	HyDe	minilm	Small	0.783	0.861	0.888	0.824	0.783	0.830	0.840

that IVF benefits from reranking more on smaller datasets. BGE reranking with Fusion also performs competitively, showing a steady gain in recall and nDCG as the dataset size decreases, peaking at R1 = 0.548 and nDCG10 = 0.740 for the small dataset. Hierarchical retrievers show moderate improvements, particularly with minilm reranking, achieving R1 = 0.608 and MRR = 0.639 on the small dataset. HyDe retrievers also demonstrate strong performance, especially when paired with minilm reranking, consistently achieving R1 above 0.72 across dataset sizes. Overall, IVF with Cosine shows a clear benefit from reranking, and minilm emerges as the most effective reranker for improving top-rank accuracy and overall ranking quality (MRR, nDCG).

Table 16 presents the reranking performance of IVF with Cosine similarity. Fusion-based retrieval with minilm reranking consistently achieves the strongest results across all dataset sizes, with R1 increasing from 0.760 (large) to 0.818 (small) and MRR reaching 0.862, highlighting that IVF benefits from reranking more on smaller datasets. BGE reranking with Fusion also performs competitively, showing a steady gain in recall and nDCG as the dataset size decreases, peaking at R1 = 0.548 and nDCG10 = 0.740 for the small dataset. Hierarchical retrievers show moderate improvements, particularly with minilm reranking, achieving R1 = 0.608 and MRR = 0.639 on the small dataset. HyDe retrievers also demonstrate strong performance, especially when paired with minilm reranking, consistently achieving R1 above 0.72 across dataset sizes. Overall, IVF with Cosine shows a clear benefit from reranking, and minilm emerges as the most effective reranker for improving top-rank accuracy and overall ranking quality (MRR, nDCG).

Table 18 presents the performance of IVF with Euclidean distance (L2) under various retrievers, rerankers, and dataset sizes. Overall, L2-based reranking yields noticeably weaker

TABLE 16. Reranking performance of IVF with Cosine across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
IVF	Cosine	Fusion	BGE	Large	0.522	0.748	0.870	0.645	0.522	0.654	0.700
IVF	Cosine	Fusion	BGE	Medium	0.531	0.762	0.891	0.660	0.531	0.667	0.717
IVF	Cosine	Fusion	BGE	Small	0.548	0.783	0.927	0.679	0.548	0.685	0.740
IVF	Cosine	Fusion	minilm	Large	0.760	0.842	0.870	0.804	0.760	0.809	0.820
IVF	Cosine	Fusion	minilm	Medium	0.786	0.866	0.891	0.828	0.786	0.834	0.844
IVF	Cosine	Fusion	minilm	Small	0.818	0.901	0.927	0.862	0.818	0.868	0.878
IVF	Cosine	Hierarchical	BGE	Large	0.307	0.438	0.535	0.381	0.307	0.382	0.418
IVF	Cosine	Hierarchical	BGE	Medium	0.346	0.495	0.620	0.436	0.346	0.433	0.481
IVF	Cosine	Hierarchical	BGE	Small	0.389	0.553	0.688	0.486	0.389	0.484	0.535
IVF	Cosine	Hierarchical	minilm	Large	0.466	0.521	0.535	0.494	0.466	0.499	0.504
IVF	Cosine	Hierarchical	minilm	Medium	0.543	0.603	0.620	0.574	0.543	0.579	0.585
IVF	Cosine	Hierarchical	minilm	Small	0.608	0.668	0.688	0.639	0.608	0.644	0.651
IVF	Cosine	HyDe	BGE	Large	0.472	0.670	0.828	0.591	0.472	0.588	0.648
IVF	Cosine	HyDe	BGE	Medium	0.483	0.698	0.859	0.607	0.483	0.608	0.668
IVF	Cosine	HyDe	BGE	Small	0.491	0.695	0.870	0.614	0.491	0.610	0.676
IVF	Cosine	HyDe	minilm	Large	0.720	0.797	0.828	0.761	0.720	0.766	0.777
IVF	Cosine	HyDe	minilm	Medium	0.749	0.828	0.859	0.792	0.749	0.797	0.808
IVF	Cosine	HyDe	minilm	Small	0.768	0.843	0.870	0.807	0.768	0.813	0.823

TABLE 17. Reranking performance of IVF with Inner Product (IP) across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
IVF	IP	Fusion	BGE	Large	0.573	0.802	0.910	0.696	0.573	0.707	0.749
IVF	IP	Fusion	BGE	Medium	0.579	0.809	0.923	0.703	0.579	0.713	0.757
IVF	IP	Fusion	BGE	Small	0.574	0.809	0.937	0.703	0.574	0.711	0.760
IVF	IP	Fusion	minilm	Large	0.796	0.879	0.910	0.841	0.796	0.846	0.858
IVF	IP	Fusion	minilm	Medium	0.815	0.898	0.923	0.859	0.815	0.865	0.875
IVF	IP	Fusion	minilm	Small	0.823	0.910	0.937	0.869	0.823	0.876	0.886
IVF	IP	Hierarchical	BGE	Large	0.318	0.463	0.567	0.399	0.318	0.401	0.440
IVF	IP	Hierarchical	BGE	Medium	0.343	0.507	0.632	0.438	0.343	0.438	0.485
IVF	IP	Hierarchical	BGE	Small	0.463	0.677	0.838	0.587	0.463	0.588	0.648
IVF	IP	Hierarchical	minilm	Large	0.493	0.552	0.567	0.522	0.493	0.528	0.533
IVF	IP	Hierarchical	minilm	Medium	0.547	0.613	0.632	0.581	0.547	0.587	0.593
IVF	IP	Hierarchical	minilm	Small	0.743	0.818	0.838	0.781	0.743	0.788	0.795
IVF	IP	HyDe	BGE	Large	0.475	0.673	0.832	0.594	0.475	0.592	0.651
IVF	IP	HyDe	BGE	Medium	0.483	0.692	0.858	0.607	0.483	0.605	0.667
IVF	IP	HyDe	BGE	Small	0.497	0.707	0.880	0.622	0.497	0.620	0.684
IVF	IP	HyDe	minilm	Large	0.723	0.800	0.832	0.764	0.723	0.769	0.781
IVF	IP	HyDe	minilm	Medium	0.749	0.828	0.858	0.791	0.749	0.796	0.808
IVF	IP	HyDe	minilm	Small	0.776	0.852	0.880	0.816	0.776	0.822	0.832

results compared to Inner Product (IP) and Cosine similarity, especially for Fusion with BGE, where R1 drops below 0.26 even on the largest dataset. MiniLM reranking improves performance slightly but remains below 0.35 in R1 for Fusion, indicating that L2 is suboptimal for dense embedding similarity in this setting. In contrast, Hierarchical retrievers achieve relatively better results under L2, particularly with MiniLM, where R1 increases from 0.485 (large) to 0.618 (small), suggesting that L2 may complement hierarchical clustering–based retrieval to some extent. HyDe retrievers paired with MiniLM consistently deliver the strongest results in this configuration, achieving R1 of up to 0.753 and MRR of nearly 0.794 on the small dataset, which narrows the gap with IP and Cosine. nDCG metrics follow the same pattern, confirming that improvements occur not just in recall but also in ranking quality. These findings suggest that while L2 is generally less effective for embedding-based retrieval, combining it with HyDe and MiniLM reranking can partially mitigate performance loss, making it viable in specific scenarios where L2 is required due to computational or hardware constraints.

Table 19 reports the reranking performance of SCaNN with Cosine similarity across retrievers, rerankers, and dataset scales. Similar to IVF, Fusion with MiniLM consistently yields the strongest outcomes, achieving R1 = 0.810 and MRR = 0.855 on the small dataset, with only marginal drops on larger datasets. Fusion with BGE performs moderately well, reaching R1 around 0.534–0.542, while still benefitting from larger datasets and showing a steady increase in nDCG values. Hierarchical retrievers are weaker overall, especially with BGE (R1 = 0.317 on the large dataset), though MiniLM reranking provides a substantial boost, raising R1 to 0.673 on the small dataset. HyDe retrievers again demonstrate competitive performance, with MiniLM pushing results to

TABLE 18. Reranking performance of IVF with Euclidean distance (L2) across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
IVF	l2	Fusion	BGE	Large	0.258	0.330	0.370	0.298	0.258	0.300	0.316
IVF	l2	Fusion	BGE	Medium	0.227	0.288	0.323	0.261	0.227	0.262	0.276
IVF	l2	Fusion	BGE	Small	0.188	0.239	0.271	0.218	0.188	0.219	0.231
IVF	l2	Fusion	minilm	Large	0.334	0.363	0.370	0.349	0.334	0.352	0.354
IVF	l2	Fusion	minilm	Medium	0.297	0.317	0.323	0.307	0.297	0.309	0.311
IVF	l2	Fusion	minilm	Small	0.247	0.266	0.271	0.256	0.247	0.258	0.260
IVF	l2	Hierarchical	BGE	Large	0.317	0.455	0.556	0.396	0.317	0.397	0.434
IVF	l2	Hierarchical	BGE	Medium	0.358	0.518	0.638	0.450	0.358	0.450	0.495
IVF	l2	Hierarchical	BGE	Small	0.394	0.566	0.698	0.494	0.394	0.494	0.544
IVF	l2	Hierarchical	minilm	Large	0.485	0.540	0.556	0.513	0.485	0.518	0.524
IVF	l2	Hierarchical	minilm	Medium	0.557	0.619	0.638	0.589	0.557	0.594	0.601
IVF	l2	Hierarchical	minilm	Small	0.618	0.678	0.698	0.649	0.618	0.654	0.661
IVF	l2	HyDe	BGE	Large	0.474	0.672	0.822	0.591	0.474	0.590	0.647
IVF	l2	HyDe	BGE	Medium	0.480	0.688	0.849	0.601	0.480	0.601	0.661
IVF	l2	HyDe	BGE	Small	0.485	0.693	0.857	0.607	0.485	0.606	0.667
IVF	l2	HyDe	minilm	Large	0.718	0.791	0.822	0.757	0.718	0.761	0.773
IVF	l2	HyDe	minilm	Medium	0.743	0.818	0.849	0.784	0.743	0.788	0.800
IVF	l2	HyDe	minilm	Small	0.753	0.831	0.857	0.794	0.753	0.800	0.809

R1 = 0.776 and MRR = 0.815 on small datasets, rivalling Fusion. Across all settings, nDCG scores closely follow recall trends, reinforcing that improvements reflect not only higher retrieval coverage but also better placement of relevant documents in the top ranks. These results indicate that SCaNN with Cosine, particularly when paired with MiniLM reranking, is highly effective, providing robust performance across retrievers and dataset sizes.

Table 20 presents the reranking performance of SCaNN with Inner Product across retrievers, rerankers, and dataset scales. Fusion with MiniLM emerges as the most effective combination, achieving the strongest results across all dataset sizes, with R1 values reaching 0.828 and MRR = 0.872 on the small dataset, and maintaining consistently high performance even on larger sets. Fusion with BGE performs moderately well (R1 = 0.573–0.578), showing incremental gains in R3 and R10 as the dataset size decreases. Hierarchical retrievers underperform with BGE, particularly on large datasets (R1 = 0.309). However, reranking with MiniLM substantially improves the results, boosting R1 to 0.738 on small datasets. HyDe retrievers demonstrate balanced performance, with MiniLM reranking producing competitive scores (R1 = 0.781 and MRR = 0.821 on small datasets), closely approaching the performance of Fusion. Across all settings, nDCG trends mirror recall improvements, indicating that gains are not only due to higher coverage but also better ranking of relevant items. Overall, SCaNN with IP, coupled with MiniLM, provides a robust and reliable reranking pipeline that outperforms other retriever–reranker combinations, especially on smaller datasets.

Table 21 reports the reranking performance of SCaNN with Euclidean Distance (L2) across different retrievers, rerankers, and dataset sizes. The results show a clear performance gap depending on the retriever–reranker combination. Fusion retrievers with either BGE or MiniLM yield comparatively low R1 values ($\leq$0.32) and shallow improvements in R3 and R10, suggesting weak alignment between L2 similarity and semantic relevance. Hierarchical retrievers perform moderately better, with gains from small dataset sizes (e.g., R1 = 0.620 for MiniLM) showing more substantial precision at top ranks, though performance remains limited at larger scales. In contrast, HyDe retrievers paired with MiniLM consistently achieve the best results across dataset sizes, with R1 above 0.72 and stable nDCG values, highlighting their effectiveness in leveraging L2 for semantic matching. Notably, HyDe + MiniLM small-scale configurations achieve the highest overall performance (R1 = 0.762, R10 = 0.865, MRR = 0.801). These findings indicate that while SCaNN with L2 underperforms in Fusion setups, its synergy with HyDe–MiniLM offers competitive retrieval quality, especially in smaller-scale collections.

C. COMPARATIVE ANALYSIS

Fig. 2 shows the comparative analysis of HNSW, IVF, and SCaNN across performance, latency, cost, and token metrics. It reveals that all three indices achieve nearly identical outcomes in terms of coverage, correctness, faithfulness, relevance, token usage, and cost, indicating no significant trade-offs in retrieval quality or resource consumption. The main point of divergence lies in latency: SCaNN achieves the lowest mean latency (3.05 ns) and p95 latency (4.44 ns), followed closely by IVF, while HNSW incurs higher delays (3.50 ns mean, 5.03 ns p95). This demonstrates that although the indices are equivalent in reliability and efficiency from a quality and cost standpoint, SCaNN provides superior retrieval speed, making it the most suitable choice for latency-sensitive applications.

As shown in Fig. 3, the analysis of dataset size (Large, Medium, Small) indicates that retrieval quality metrics such as coverage, correctness, faithfulness, and relevance remain stable across different sizes, with only marginal improvements for smaller datasets. For example, correctness rises

TABLE 19. Reranking performance of SCaNN with Cosine across different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
SCANN	Cosine	Fusion	BGE	Large	0.534	0.765	0.885	0.660	0.534	0.669	0.715
SCANN	Cosine	Fusion	BGE	Medium	0.534	0.766	0.895	0.664	0.534	0.671	0.720
SCANN	Cosine	Fusion	BGE	Small	0.542	0.783	0.923	0.673	0.542	0.681	0.734
SCANN	Cosine	Fusion	minilm	Large	0.778	0.858	0.885	0.820	0.778	0.826	0.836
SCANN	Cosine	Fusion	minilm	Medium	0.784	0.869	0.895	0.829	0.784	0.836	0.845
SCANN	Cosine	Fusion	minilm	Small	0.810	0.898	0.923	0.855	0.810	0.863	0.872
SCANN	Cosine	Hierarchical	BGE	Large	0.317	0.468	0.578	0.403	0.317	0.405	0.446
SCANN	Cosine	Hierarchical	BGE	Medium	0.358	0.520	0.642	0.451	0.358	0.452	0.497
SCANN	Cosine	Hierarchical	BGE	Small	0.422	0.611	0.759	0.533	0.422	0.532	0.588
SCANN	Cosine	Hierarchical	minilm	Large	0.501	0.558	0.578	0.531	0.501	0.535	0.542
SCANN	Cosine	Hierarchical	minilm	Medium	0.565	0.625	0.642	0.595	0.565	0.601	0.607
SCANN	Cosine	Hierarchical	minilm	Small	0.673	0.738	0.759	0.707	0.673	0.712	0.720
SCANN	Cosine	HyDe	BGE	Large	0.473	0.675	0.830	0.593	0.473	0.592	0.650
SCANN	Cosine	HyDe	BGE	Medium	0.482	0.693	0.852	0.605	0.482	0.605	0.665
SCANN	Cosine	HyDe	BGE	Small	0.496	0.701	0.877	0.619	0.496	0.616	0.682
SCANN	Cosine	HyDe	minilm	Large	0.723	0.798	0.830	0.763	0.723	0.768	0.779
SCANN	Cosine	HyDe	minilm	Medium	0.743	0.819	0.852	0.785	0.743	0.789	0.801
SCANN	Cosine	HyDe	minilm	Small	0.776	0.849	0.877	0.815	0.776	0.820	0.830

TABLE 20. Reranking performance of SCaNN with Inner Product (IP) different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
SCANN	IP	Fusion	BGE	Large	0.578	0.803	0.911	0.699	0.578	0.710	0.752
SCANN	IP	Fusion	BGE	Medium	0.578	0.812	0.925	0.702	0.578	0.713	0.757
SCANN	IP	Fusion	BGE	Small	0.573	0.807	0.937	0.702	0.573	0.710	0.759
SCANN	IP	Fusion	minilm	Large	0.798	0.881	0.911	0.842	0.798	0.848	0.859
SCANN	IP	Fusion	minilm	Medium	0.819	0.901	0.925	0.862	0.819	0.868	0.878
SCANN	IP	Fusion	minilm	Small	0.828	0.910	0.937	0.872	0.828	0.878	0.888
SCANN	IP	Hierarchical	BGE	Large	0.309	0.450	0.559	0.391	0.309	0.391	0.432
SCANN	IP	Hierarchical	BGE	Medium	0.358	0.522	0.643	0.452	0.358	0.453	0.498
SCANN	IP	Hierarchical	BGE	Small	0.456	0.666	0.831	0.579	0.456	0.578	0.640
SCANN	IP	Hierarchical	minilm	Large	0.484	0.543	0.559	0.514	0.484	0.520	0.525
SCANN	IP	Hierarchical	minilm	Medium	0.558	0.623	0.643	0.592	0.558	0.597	0.604
SCANN	IP	Hierarchical	minilm	Small	0.738	0.809	0.831	0.775	0.738	0.781	0.789
SCANN	IP	HyDe	BGE	Large	0.476	0.673	0.833	0.594	0.476	0.592	0.652
SCANN	IP	HyDe	BGE	Medium	0.482	0.692	0.851	0.604	0.482	0.604	0.664
SCANN	IP	HyDe	BGE	Small	0.496	0.713	0.884	0.624	0.496	0.623	0.687
SCANN	IP	HyDe	minilm	Large	0.723	0.802	0.833	0.765	0.723	0.770	0.781
SCANN	IP	HyDe	minilm	Medium	0.746	0.823	0.851	0.787	0.746	0.793	0.803
SCANN	IP	HyDe	minilm	Small	0.781	0.857	0.884	0.821	0.781	0.826	0.837

slightly from 0.73 (Large) to 0.78 (Small), and faithfulness reaches its peak at 0.93 for Small. Latency shows minimal variation across sizes, with a mean latency of 3.2ns and p95 latency between 4.65–4.69ns, indicating consistent retrieval speed regardless of scale. Cost metrics are virtually identical, with mean cost fixed at $1.2e^{-4}$ and slight variations in standard deviation. Token usage also remains stable, though smaller datasets exhibit slightly lower prompt token means (546 vs. 558) and total token counts (612 vs. 625).

As illustrated in Fig. 4, comparison across similarity metrics (Cosine, Inner Product, and Euclidean L2) reveals more noticeable differences. Cosine and IP consistently deliver higher coverage (0.81–0.82) and correctness (0.79–0.80) compared to L2, which lags with values around 0.60 for coverage and 0.67 for correctness. Faithfulness remains strong and uniform (0.91–0.93), while relevance is highest for Cosine and IP (0.87) compared to L2 (0.78). Latency again shows only minor variation, with L2 achieving the lowest mean latency (3.07 ns) and p95 latency (4.51 ns), followed by Cosine and IP. Costs and token usage are nearly indistinguishable across metrics, reinforcing that the primary differences lie in retrieval quality rather than computational efficiency. Thus, Cosine and IP emerge as stronger options for accuracy oriented applications, while L2 offers slightly faster retrieval at the cost of reduced quality.

As shown in Fig. 5, the comparison of Fusion, Hierarchical, and HyDe retrievers reveals clear trade-offs across performance, latency, cost, and token usage. HyDe consistently delivers the best retrieval quality, achieving the highest coverage (0.86), correctness, faithfulness, and relevance (0.89), but this comes at the expense of efficiency, with the slowest mean and p95 latencies (4.84ns and 6.93ns), the highest cost (1.7e-04, nearly double Fusion and Hierarchical), and the most significant token consumption (710 total tokens

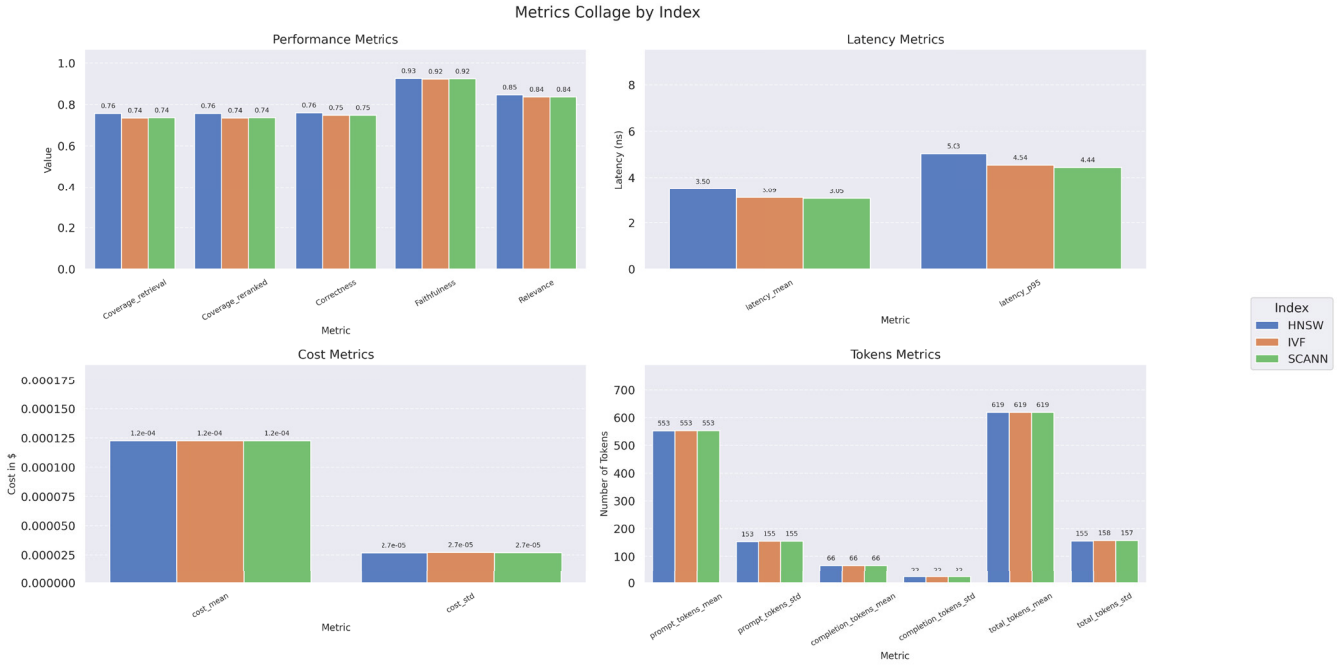

FIGURE 2. Comparative performance of HNSW, IVF, and SCaNN across key metrics including retrieval quality, latency, cost, and token efficiency.

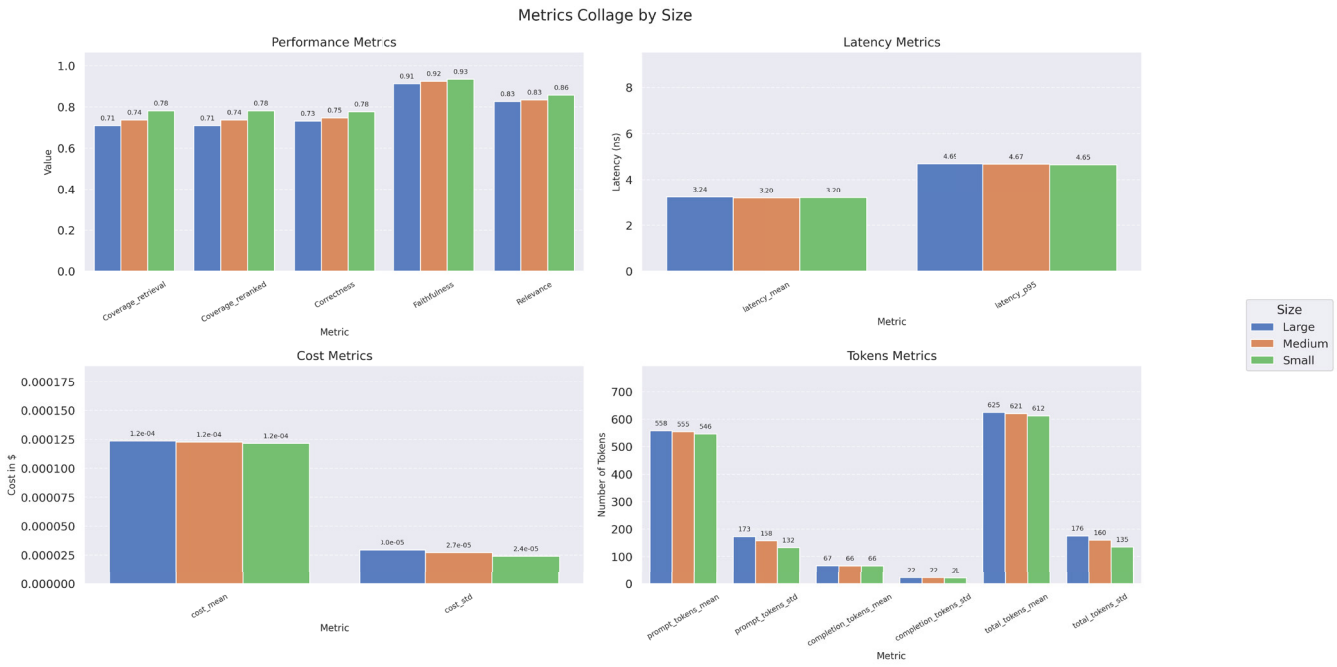

FIGURE 3. Comparative performance on Small, Medium, and Large size datasets across key metrics including retrieval quality, latency, cost, and token efficiency.

on average). In contrast, Hierarchical is the fastest and most resource-efficient, with a mean latency of 2.21 ns, a p95 latency of 3.11 ns, and lower cost and token usage (571 total tokens). However, its accuracy and coverage lag (0.65–0.78). Fusion occupies a middle ground, balancing accuracy, speed, and efficiency with moderate values across all metrics. Retriever choice depends on application needs: HyDe for accuracy-critical tasks, Hierarchical for latency-sensitive scenarios, and Fusion for balanced performance.

Fig. 6 compares the rerankers BGE and minilm, showing only minor differences in performance, latency, cost, and token usage. Both models achieve identical scores in

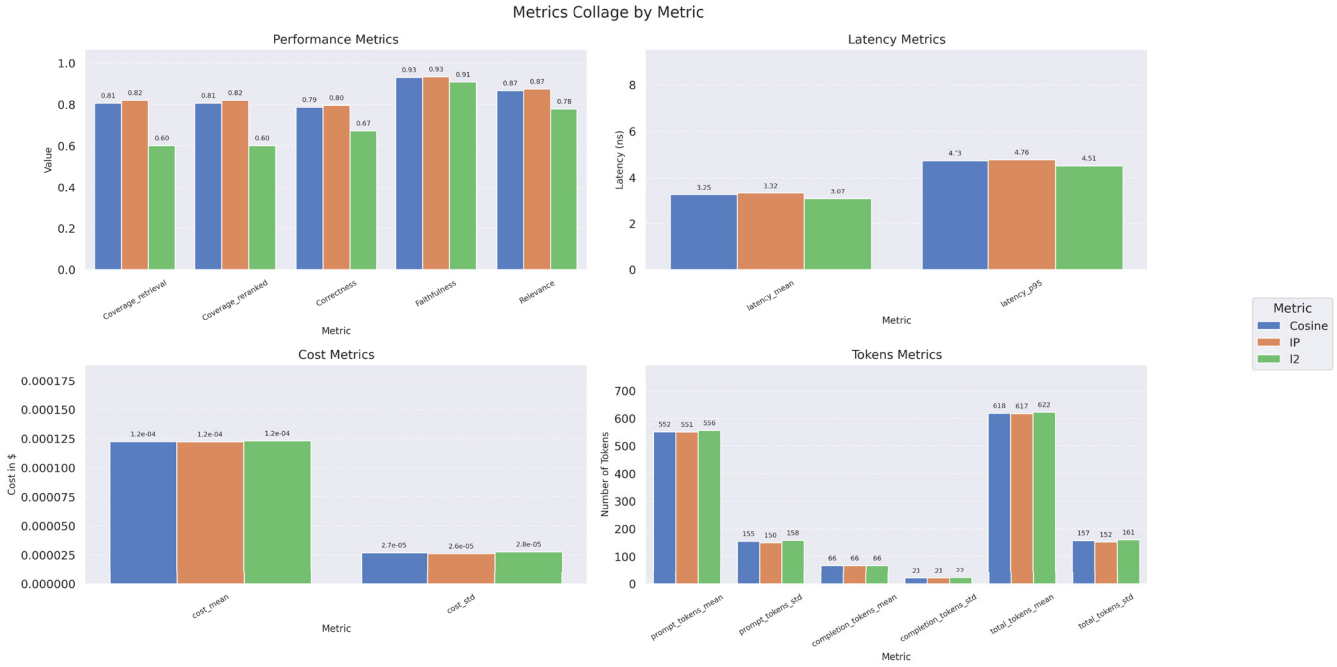

FIGURE 4. Comparative performance of Cosine, Inner Product and L2 metrics across key metrics including retrieval quality, latency, cost, and token efficiency.

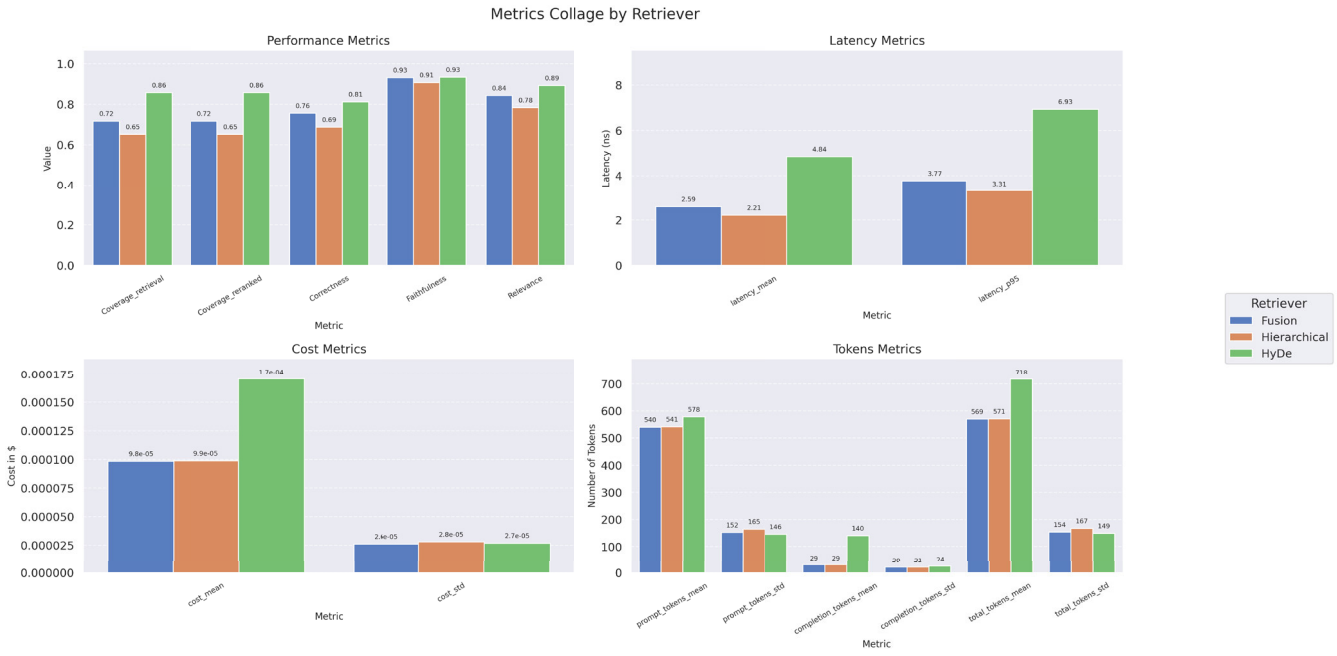

FIGURE 5. Comparative performance of Fusion, Hierarchical, and HyDe metrics across key metrics including retrieval quality, latency, cost, and token efficiency.

Coverage Retrieval and Coverage Reranked (0.74). At the same time, minilm performs slightly better in Correctness (0.78 vs. 0.73) and Faithfulness (0.93 vs. 0.92), and BGE shows a marginally higher Relevance score (0.86 vs. 0.82), suggesting near-equivalent retrieval quality. Latency provides the most evident distinction, with minilm being faster (mean 3.07s, p95 4.61 ns) than BGE (mean 3.35 ns, p95 4.73 ns), making minilm more suitable for speed-sensitive applications; however, the gap is modest compared to retrievers. Cost and token metrics reveal almost no difference, with both rerankers averaging a mean cost of 1.2e-04 and similar token usage (616 for BGE vs. 622 for minilm). Overall, reranker

TABLE 21. Reranking performance of SCaNN with Euclidean Distance (L2) different retrievers, rerankers, and dataset sizes. Metrics include Recall@1, @3, @10 (R1, R3 and R10, fraction of queries with correct item in top-K), MRR (average reciprocal rank of first correct result), and nDCG@1, @3, @10 (nDCG1, nDCG3, nDCG10 account for relevance and position of retrieved items).

Index	Metric	Retriever	Reranker	Size	R1	R3	R10	MRR	nDCG1	nDCG3	nDCG10
SCANN	l2	Fusion	BGE	Large	0.248	0.323	0.354	0.287	0.248	0.291	0.304
SCANN	l2	Fusion	BGE	Medium	0.223	0.290	0.323	0.260	0.223	0.263	0.276
SCANN	l2	Fusion	BGE	Small	0.182	0.232	0.267	0.211	0.182	0.211	0.224
SCANN	l2	Fusion	minilm	Large	0.323	0.348	0.354	0.335	0.323	0.338	0.340
SCANN	l2	Fusion	minilm	Medium	0.299	0.318	0.323	0.309	0.299	0.310	0.313
SCANN	l2	Fusion	minilm	Small	0.241	0.259	0.267	0.251	0.241	0.252	0.255
SCANN	l2	Hierarchical	BGE	Large	0.291	0.408	0.506	0.361	0.291	0.360	0.396
SCANN	l2	Hierarchical	BGE	Medium	0.318	0.446	0.558	0.396	0.318	0.393	0.435
SCANN	l2	Hierarchical	BGE	Small	0.397	0.559	0.704	0.497	0.397	0.492	0.547
SCANN	l2	Hierarchical	minilm	Large	0.442	0.493	0.506	0.468	0.442	0.473	0.477
SCANN	l2	Hierarchical	minilm	Medium	0.486	0.540	0.558	0.515	0.486	0.519	0.526
SCANN	l2	Hierarchical	minilm	Small	0.620	0.683	0.704	0.653	0.620	0.658	0.666
SCANN	l2	HyDe	BGE	Large	0.477	0.675	0.829	0.593	0.477	0.593	0.651
SCANN	l2	HyDe	BGE	Medium	0.478	0.682	0.845	0.599	0.478	0.597	0.658
SCANN	l2	HyDe	BGE	Small	0.491	0.698	0.865	0.613	0.491	0.611	0.674
SCANN	l2	HyDe	minilm	Large	0.723	0.800	0.829	0.764	0.723	0.769	0.780
SCANN	l2	HyDe	minilm	Medium	0.742	0.814	0.845	0.781	0.742	0.785	0.797
SCANN	l2	HyDe	minilm	Small	0.762	0.838	0.865	0.801	0.762	0.807	0.817

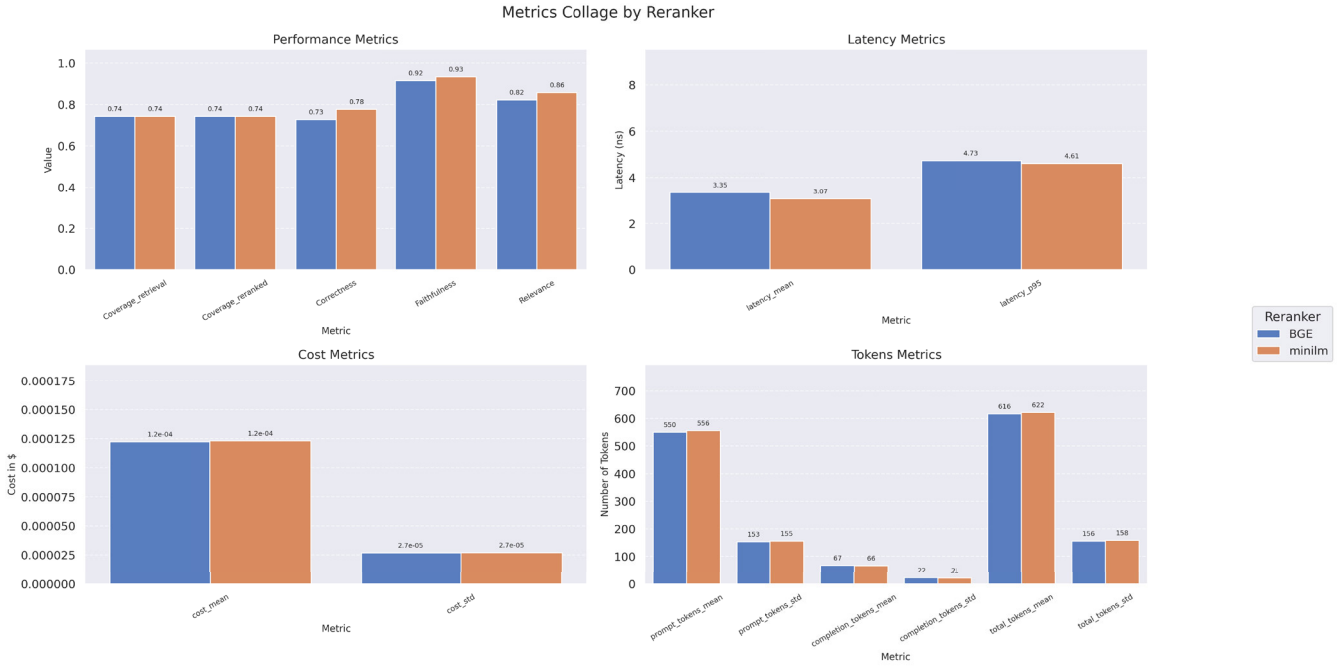

FIGURE 6. Comparative performance of BGE and minilm metrics across key metrics, including retrieval quality, latency, cost, and token efficiency.

selection has minimal impact on performance or resource efficiency, with only slight advantages depending on whether accuracy (minilm) or relevance (BGE) is prioritized.

Table 22 summarizes the performance, efficiency, and cost-related metrics across all retrieval and reranking configurations. Coverage metrics show a mean of 0.743 with high variability, with maximum coverage achieved using HNSW, IP, Fusion, and BGE on the Small dataset, and minimum coverage with SCANN, l2, Fusion, and BGE on the Small dataset. Correctness, Faithfulness, and Relevance exhibit high mean values (0.752–0.924), with Faithfulness being the most consistent. Maximum performance is generally observed with HNSW and minilm-based rerankers. Latency varies notably, with a mean latency of 3.212 ns and a P95 of 4.669 ns, peaking for HNSW, IP, Fusion, minilm, and Small combinations, reflecting the trade-off between accuracy and computational cost. Cost and token usage remain generally low and efficient, although the total number of tokens ranges from 553 to 725, depending on the configuration. Overall, the Table highlights the trade-offs between retrieval accuracy, consistency, latency, and computational efficiency across different system settings.

The Table 22 shows that as retrieval datasets grow from small to medium, retrieval quality, measured by coverage,

TABLE 22. Summary statistics for all metrics, including mean, std, max/min values and their corresponding setting combinations.

Metric	Mean	Std	Max	Max Combination	Min	Min Combination
Coverage Retrieval	0.743	0.193	0.942	HNSW, IP, Fusion, BGE, Small	0.267	SCANN, l2, Fusion, BGE, Small
Coverage Reranked	0.743	0.193	0.942	HNSW, IP, Fusion, BGE, Small	0.267	SCANN, l2, Fusion, BGE, Small
Correctness	0.752	0.114	0.909	HNSW, IP, Fusion, minilm, Small	0.484	SCANN, l2, Fusion, BGE, Small
Faithfulness	0.924	0.026	0.970	HNSW, IP, Fusion, minilm, Small	0.875	SCANN, IP, Hierarchical, BGE, Large
Relevance	0.840	0.091	0.959	HNSW, IP, Fusion, minilm, Small	0.615	SCANN, l2, Fusion, BGE, Small
Mean latency	3.212	1.336	6.585	HNSW, IP, Fusion, minilm, Small	1.736	IVF, l2, Hierarchical, minilm, Medium
P95 latency	4.669	1.802	8.673	HNSW, IP, Fusion, minilm, Small	2.574	IVF, l2, Hierarchical, minilm, Medium
Mean Cost	1.230e-04	3.400e-05	1.720e-04	SCANN, IP, HyDe, minilm, Large	9.600e-05	IVF, Cosine, Hierarchical, BGE, Small
Cost std	2.700e-05	3.000e-06	3.900e-05	IVF, Cosine, Hierarchical, BGE, Large	2.300e-05	HNSW, Cosine, Fusion, minilm, Small
Mean prompt tokens	552.935	19.392	584.246	IVF, Cosine, HyDe, minilm, Large	523.263	SCANN, Cosine, Hierarchical, BGE, Small
Prompt tokens std	154.272	24.068	244.002	IVF, Cosine, Hierarchical, BGE, Large	126.973	IVF, Cosine, Fusion, BGE, Small
Mean completion tokens	66.264	52.432	141.109	SCANN, l2, HyDe, BGE, Large	26.977	SCANN, l2, Fusion, BGE, Small
Completion tokens std	21.681	2.152	25.058	SCANN, l2, HyDe, BGE, Large	18.263	SCANN, IP, Fusion, minilm, Small
Mean total tokens	619.199	70.429	724.878	SCANN, IP, HyDe, minilm, Large	552.645	SCANN, Cosine, Hierarchical, BGE, Small
Total tokens std	156.756	23.862	245.468	IVF, Cosine, Hierarchical, BGE, Large	129.591	HNSW, Cosine, HyDe, BGE, Small

correctness, and faithfulness, improves due to greater document diversity and a higher likelihood of finding relevant information. However, these benefits largely plateau from medium to large datasets, suggesting a saturation point where adding more documents introduces more noise than signal, especially in SCANN and L2-based settings, which show the weakest scalability.

As shown in Table 23, the best-performing parameter combinations vary across retrieval, reranking, correctness, efficiency, and token-related metrics. The table systematically compares different retrievers (HNSW, IVF, SCANN), similarity functions (IP, Cosine, L2), reranking strategies (Fusion, Hierarchical, HyDe), and rerankers (BGE, minilm). These results highlight clear performance trade-offs, where specific configurations excel in retrieval precision, while others optimize computational efficiency or semantic quality.

D. GUIDING PRINCIPLES

For retrieval-level metrics, HNSW combined with IP similarity and Fusion reranking shows strong performance, particularly with the BGE reranker. It achieves the highest R@3 (0.887), R@10 (0.942), and Coverage (0.942), demonstrating its effectiveness in deeper recall. Meanwhile, IVF with IP–Fusion–BGE yields the strongest R@1 and nDCG@1 scores (both 0.752), suggesting that IVF may be more precise in top-ranked retrieval compared to HNSW. This distinction underlines the importance of retriever choice depending on whether the task prioritizes breadth (coverage) or top precision.

At the reranked stage, the influence of the reranker becomes more pronounced. SCANN with IP–Fusion–minilm delivers the best R@1 reranked (0.828) and nDCG@1 reranked (0.828), while HNSW with minilm achieves the strongest R@3 reranked (0.913), MRR reranked (0.872), and nDCG@10 reranked (0.889). Notably, the correctness (0.909), faithfulness (0.970), and relevance (0.959) metrics peak under HNSW–IP–Fusion with minilm, indicating that reranking with minilm substantially enhances semantic accuracy and trustworthiness beyond raw retrieval.

For efficiency related metrics, IVF in combination with hierarchical reranking and minilm provides the lowest latencies, reaching a mean of 1.736s and a p95 latency of 2.574s. Cost metrics also favor lightweight configurations: IVF–Cosine–Hierarchical–BGE minimizes average cost (0.000096), while HNSW–Cosine–Fusion–minilm reduces cost variance (0.000023). These results suggest that IVF configurations are preferable in latency-sensitive or resource-constrained scenarios.

Finally, token-level metrics in Table 23 show distinct patterns: SCANN–Cosine–Hierarchical–BGE maximizes prompt (523.263) and total tokens (552.645), whereas SCANN–l2–Fusion–BGE yields the highest completion token mean (26.977). Variance measures indicate stability differences, with SCANN–IP–Fusion–minilm minimizing completion token variability (18.263). These results

TABLE 23. Best performing parameter combinations for each metric.

Performance Metric	Index	Metric	Retriever	Reranker	Size	Best Value
R@1_retrieval	IVF	IP	Fusion	BGE	Small	0.752000
R@3_retrieval	HNSW	IP	Fusion	BGE	Small	0.887000
R@10_retrieval	HNSW	IP	Fusion	BGE	Small	0.942000
MRR_retrieval	HNSW	IP	Fusion	BGE	Small	0.823000
nDCG@1_retrieval	IVF	IP	Fusion	BGE	Small	0.752000
nDCG@3_retrieval	HNSW	IP	Fusion	BGE	Small	0.832000
nDCG@10_retrieval	HNSW	IP	Fusion	BGE	Small	0.852000
Coverage_retrieval	HNSW	IP	Fusion	BGE	Small	0.942000
R@1_reranked	SCANN	IP	Fusion	minilm	Small	0.828000
R@3_reranked	HNSW	IP	Fusion	minilm	Small	0.913000
R@10_reranked	HNSW	IP	Fusion	BGE	Small	0.942000
MRR_reranked	HNSW	IP	Fusion	minilm	Small	0.872000
nDCG@1_reranked	SCANN	IP	Fusion	minilm	Small	0.828000
nDCG@3_reranked	HNSW	IP	Fusion	minilm	Small	0.878000
nDCG@10_reranked	HNSW	IP	Fusion	minilm	Small	0.889000
Coverage_reranked	HNSW	IP	Fusion	BGE	Small	0.942000
Correctness	HNSW	IP	Fusion	minilm	Small	0.909000
Faithfulness	HNSW	IP	Fusion	minilm	Small	0.970000
Relevance	HNSW	IP	Fusion	minilm	Small	0.959000
latency_mean	IVF	l2	Hierarchical	minilm	Medium	1.736000
latency_p95	IVF	l2	Hierarchical	minilm	Medium	2.574000
cost_mean	IVF	Cosine	Hierarchical	BGE	Small	0.000096
cost_std	HNSW	Cosine	Fusion	minilm	Small	0.000023
prompt_tokens_mean	SCANN	Cosine	Hierarchical	BGE	Small	523.263
prompt_tokens_std	IVF	Cosine	Fusion	BGE	Small	126.973
completion_tokens_mean	SCANN	l2	Fusion	BGE	Small	26.977
completion_tokens_std	SCANN	IP	Fusion	minilm	Small	18.263
total_tokens_mean	SCANN	Cosine	Hierarchical	BGE	Small	552.645
total_tokens_std	HNSW	Cosine	HyDe	BGE	Small	129.591

demonstrate that token usage is significantly influenced by the retriever–reranker choice and may reflect the linguistic richness of the retrieved content.

VI. LIMITATION

Although the EVARAG framework provides a systematic and comprehensive framework for benchmarking RAG systems, several limitations remain. The experimental evaluation was conducted on a restricted set of QA dataset (SquAD), which doesn't fully capture the diversity, particularly in specialized domains such as medicine, law, or finance. Similarly, while our metrics focused on retrieval accuracy and generative quality, broader aspects such as factual consistency, bias, and robustness were not extensively measured. Another important consideration is that although chunking strategies were considered, adaptive or dynamic chunking approaches that could significantly affect retrieval quality were not fully incorporated. Also, our system does not account for the case where feature boundaries are blurred, which typically occurs under high semantic density. This overlap makes it difficult for the model to clearly separate features, leading to reduced precision in downstream tasks.

VII. FUTURE WORK

Future research will extend EVARAG in several promising directions. One natural step is to evaluate RAG pipelines in domain-specific contexts, such as healthcare or legal applications, to assess the robustness of retriever-similarity-metric combinations under domain constraints. Another avenue lies in the design of hybrid similarity functions that adaptively combine cosine, inner product, and L2 distances, thereby balancing semantic precision with recall. Beyond performance, future extensions of EVARAG could incorporate metrics that explicitly measure factual grounding and hallucination reduction, providing a more holistic assessment of RAG systems. Additionally, a potential direction for future improvement is to integrate techniques such as context-aware feature disentanglement and density-adaptive representations, which can be adapted to handle overlapping feature boundaries. While our study primarily focuses on retrieval metrics, it does not explicitly examine how factors such as semantic sparsity and the distribution of embedding vectors affect retrieval efficiency. Future research will also incorporate causal analysis to investigate the impact of these factors on metric outcomes, thus improving theoretical and practical understanding of RAG systems.

VIII. CONCLUSION AND DISCUSSION

This study presented a comprehensive evaluation of RAG pipelines, examining the interplay between retrievers, similarity metrics, indexing strategies, rerankers, and dataset sizes across a wide range of performance and efficiency metrics. Our results demonstrate that component choice has a substantial impact on retrieval quality, semantic accuracy, and computational efficiency, often revealing trade-offs among these dimensions. In terms of retrieval performance, the combination of HNSW indexing with inner product (IP) similarity, Fusion reranking, and the BGE

reranker consistently delivered top-tier results on the Small dataset, achieving a maximum Coverage Retrieval of 0.942, R@10 of 0.942, and MRR of 0.823. When reranking is applied, the HNSW–IP–Fusion–minilm configuration further improved semantic quality, reaching the highest Correctness (0.909), Faithfulness (0.970), and Relevance (0.959) scores, underscoring the effectiveness of cross-encoder rerankers for semantic refinement. The effectiveness of HNSW–IP–Fusion–MiniLM stems from combining HNSW's efficient nearest neighbor search, Inner Product similarity's alignment with embedding semantics, Fusion's robust evidence aggregation, and MiniLM's lightweight, context-aware reranking. HyDe consistently outperforms Fusion when paired with L2, as its generative query expansion yields embeddings with more stable magnitudes, allowing L2 to serve as a reliable discriminator. In contrast, Fusion's raw retrieval signals lead to greater variance in embedding norms, undermining L2 performance. From an efficiency standpoint, configurations built on IVF indexing with L2 similarity and Hierarchical retrievers proved to be the most resource efficient, with a mean latency as low as 1.736 ns and a p95 latency of 2.574 ns, making them ideal for latency-sensitive applications. Cost analysis showed that IVF–Cosine–Hierarchical–BGE minimized computational expense (9.6×10^{-5}) while HNSW–Cosine–Fusion–minilm minimized cost variability (2.3×10^{-5}). Analysis of similarity metrics revealed that the Cosine and Inner Product functions outperform L2 in retrieval quality, with coverage ranging from 0.81 to 0.82, compared to 0.60 for L2, although L2 retained a slight latency advantage. Furthermore, the HyDe retriever demonstrated superior retrieval quality (Coverage 0.86, Relevance 0.89) at the cost of higher latency (4.84 ns) and token usage (710 tokens). In comparison, Hierarchical retrievers achieved the lowest latency (2.21 ns) and cost, highlighting the fundamental trade-off between precision and efficiency. Finally, reranker analysis showed marginal differences: minilm slightly outperformed BGE in correctness (0.78 vs. 0.73) and faithfulness (0.93 vs. 0.92), while maintaining lower latency (3.07 ns vs. 3.35 ns). This suggests that reranker choice has a modest but measurable impact. To summarize, our findings show that no single configuration universally dominates; instead, optimal design depends on task requirements. HNSW–IP–Fusion–minilm is ideal for accuracy-critical applications, IVF–L2–Hierarchical–minilm excels in latency-sensitive environments, and SCaNN offers a balanced trade-off with the lowest average latency (3.05 ns) and competitive performance. Beyond empirical performance, EVARAG reveals a deeper guiding principle for RAG design. Retrieval effectiveness emerges not from any single component but from the interaction between retriever similarity geometry, indexing structure, and reranker semantic refinement. The study shows that retrieval pipelines function as tightly coupled systems in which early-stage vector search determines the coarse semantic neighbourhood, while rerankers selectively sharpen this signal by resolving fine-grained meaning. This layered interaction explains why certain combinations, such as HNSW–IP–Fusion–MiniLM, consistently outperform alternatives: they align vector-space geometry with evidence aggregation and lightweight semantic filtering. Similarly, the trade-offs observed across IVF, L2, and hierarchical configurations illustrate that efficiency and precision are governed by the structural properties of the embedding space rather than by model size alone. Taken together, EVARAG demonstrates that optimal RAG performance depends on achieving alignment across these layers: indexing, similarity, retriever type, and reranking rather than merely improving any single module in isolation.

ACKNOWLEDGMENT

Computing resources used in this work were provided by the National Center for High Performance Computing of Türkiye (UHeM). This study was part of a 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).

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 article.

ETHICS DECLARATION

Not applicable.

CLINICAL TRIAL NUMBER

Not applicable.

CONSENT TO PARTICIPATE DECLARATION

Not applicable.

CONSENT FOR PUBLICATION

Not applicable.

DATA AVAILABILITY

The data supporting the findings of this study can be obtained from the author (Harun Elkiran, email: harun.elkiran@izu.edu.tr) upon reasonable request.

REFERENCES

[1] J. Achiam et al., "GPT-4 technical report," 2023, *arXiv:2303.08774*.
[2] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "LLaMA: Open and efficient foundation language models," 2023, *arXiv:2302.13971*.
[3] D. Demszky, D. Yang, D. S. Yeager, C. J. Bryan, M. Clapper, S. Chandhok, J. C. Eichstaedt, C. A. Hecht, J. P. Jamieson, M. Johnson, M. Jones, D. Krettek-Cobb, L. C. Lai, N. JonesMitchell, D. C. Ong, C. S. Dweck, J. J. Gross, and J. W. Pennebaker, "Using large language models in psychology," *Nature Rev. Psychol.*, vol. 2, no. 11, pp. 688–701, 2023.
[4] H. Nori, N. King, S. M. McKinney, D. Carignan, and E. Horvitz, "Capabilities of GPT-4 on medical challenge problems," 2023, *arXiv:2303.13375*.
[5] M. Arslan, L. Mahdjoubi, and S. Munawar, "Driving sustainable energy transitions with a multi-source RAG-LLM system," *Energy Buildings*, vol. 324, Dec. 2024, Art. no. 114827.

[6] M. Arslan, H. Ghanem, S. Munawar, and C. Cruz, "A survey on RAG with LLMs," *Proc. Comput. Sci.*, vol. 246, pp. 3781–3790, Mar. 2024.
[7] N. Kandpal, H. Deng, A. Roberts, E. Wallace, and C. Raffel, "Large language models struggle to learn long-tail knowledge," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 15696–15707.
[8] K. Sun, Y. E. Xu, H. Zha, Y. Liu, and X. L. Dong, "Head-to-tail: How knowledgeable are large language models (LLMs)? A.K.A. will LLMs replace knowledge graphs?" 2023, *arXiv:2308.10168*.
[9] J. Li, X. Cheng, W. X. Zhao, J.-Y. Nie, and J.-R. Wen, "HaluEval: A large-scale hallucination evaluation benchmark for large language models," 2023, *arXiv:2305.11747*.
[10] V. Rawte, S. Chakraborty, A. Pathak, A. Sarkar, S. M. T. I. Tonmoy, A. Chadha, A. Sheth, and A. Das, "The troubling emergence of hallucination in large language models-an extensive definition, quantification, and prescriptive remediations," in *Proc. 2023 Conf. Empirical Methods Natural Lang. Process.*, H. Bouamor, J. Pino, and K. Bali, Eds., Singapore: Association for Computational Linguistics, Dec. 2023, pp. 2541–2573, doi: 10.18653/v1/2023.emnlp-main.155.
[11] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-RAG: Learning to retrieve, generate, and critique through self-reflection," 2023, *arXiv:2310.11511*.
[12] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-T. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 9459–9474.
[13] O. Ram, Y. Levine, I. Dalmedigos, D. Muhlgay, A. Shashua, K. Leyton-Brown, and Y. Shoham, "In-context retrieval-augmented language models," *Trans. Assoc. Comput. Linguistics*, vol. 11, pp. 1316–1331, May 2023.
[14] W. Sarah, "Boosting rag performance: A comparative study of scann and traditional vector search in large language model pipelines," Apr. 2025. [Online]. Available: https://www.researchgate.net/publication/391645756_Boosting_RAG_Performance_A_Comparative_Study_of_ScaNN_and_Traditional_Vector_Search_in_Large_Language_Model_Pipelines
[15] A. Abdallah, B. Piryani, J. Mozafari, M. Ali, and A. Jatowt, "Rankify: A comprehensive Python toolkit for retrieval, re-ranking, and retrieval-augmented generation," 2025, *arXiv:2502.02464*.
[16] S. Deshmukh and A. Bajaj, "CareerBoost: A hybrid RAG-NLP job recommendation framework," in *Proc. 8th Int. Conf. I-SMAC (IoT Social, Mobile, Anal. Cloud) (I-SMAC)*, Oct. 2024, pp. 853–858.
[17] D. Mozolevskyi and W. AlShikh, "Comparative analysis of retrieval systems in the real world," 2024, *arXiv:2405.02048*.
[18] J. Kim and D. Mahajan, "VectorLiteRAG: Latency-aware and fine-grained resource partitioning for efficient RAG," 2025, *arXiv:2504.08930*.
[19] 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.
[20] 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.
[21] Y. Ateş, A. Sayar, İ. U. Bozlar, S. Ertuğrul, and S. S. Arslan, "Semantic chunking and chain-of-thought reasoning for rag-based document processing," in *Proc. IEEE 35th Int. Workshop Mach. Learn. Signal Process. (MLSP)*, May 2025, pp. 1–6.
[22] C.-Y. Chang, Z. Jiang, V. Rakesh, M. Pan, C.-C.-M. Yeh, G. Wang, M. Hu, Z. Xu, Y. Zheng, M. Das, and N. Zou, "MAIN-RAG: Multi-agent filtering retrieval-augmented generation," in *Proc. 63rd Annu. Meeting Assoc. Comput. Linguistics*, 2025, pp. 2607–2622.
[23] J. Nian, Z. Peng, Q. Wang, and Y. Fang, "W-RAG: Weakly supervised dense retrieval in RAG for open-domain question answering," in *Proc. Int. ACM SIGIR Conf. Innov. Concepts Theories Inf. Retr. (ICTIR)*, Jul. 2025, pp. 136–146.
[24] J. Hu, Y. Zhou, and J. Wang, "Intrinsic evaluation of RAG systems for deep-logic questions," 2024, *arXiv:2410.02932*.
[25] W. Wang, J. Ma, P. Zhang, Z. Hu, Q. Jiang, and Y. Liu, "Application of multi-way recall fusion reranking based on tensor and ColBERT in RAG," in *Proc. IEEE 7th Int. Conf. Inf. Syst. Comput. Aided Educ. (ICISCAE)*, Sep. 2024, pp. 138–141.
[26] 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.
[27] A. Leto, C. Aguerrebere, I. Bhati, T. Willke, M. Tepper, and V. A. Vo, "Toward optimal search and retrieval for RAG," 2024, *arXiv:2411.07396*.
[28] H. Sun, Y. Wang, and S. Zhang, "Retrieval-augmented generation for domain-specific question answering: A case study on Pittsburgh and CMU," 2024, *arXiv:2411.13691*.
[29] 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*.
[30] I. Papadimitriou, I. Gialampoukidis, S. Vrochidis, Ioannis, and Kompatsiaris, "RAG playground: A framework for systematic evaluation of retrieval strategies and prompt engineering in RAG systems," 2024, *arXiv:2412.12322*.
[31] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," 2016, *arXiv:1606.05250*.

HARUN ELKIRAN (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 from 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, Türkiye; and a Research Fellow with Applied Science Private University, Jordan. He is the author/co-author of over 80 articles published in well-reputed journals and highly-ranked conferences. His research interests include artificial intelligence and image processing, pattern recognition, the IoT, blockchain, and data analytics. He was a Gold Medalist. He received 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 at 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*.