EncouRAGe: Evaluating RAG Local, Fast, and Reliable

Jan Strich, Adeline Scharfenberg, Chris Biemann, Martin Semmann
Universität Hamburg

Abstract

We introduce **EncouRAGe**, a comprehensive Python framework designed to streamline the development and evaluation of Retrieval-Augmented Generation (RAG) systems using Large Language Models (LLMs) and Embedding Models. EncouRAGe comprises five modular and extensible components: *Type Manifest*, *RAG Factory*, *Inference*, *Vector Store*, and *Metrics*, facilitating flexible experimentation and extensible development. The framework emphasizes **scientific reproducibility**, **diverse evaluation metrics**, and **local deployment**, enabling researchers to efficiently assess datasets within RAG workflows. This paper presents implementation details and an extensive evaluation across multiple benchmark datasets, including *25k QA pairs* and *over 51k documents*. Our results show that RAG still underperforms compared to the *Oracle Context*, while *Hybrid BM25* consistently achieves the best results across all four datasets. We further examine the effects of reranking, observing only marginal performance improvements accompanied by higher response latency.

Keywords: Retrieval-Augmented Generation, Evaluation, Library

1. Introduction

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020; Huang and Huang, 2024; Nikishina et al., 2025) has emerged as a dominant approach for enhancing large language models (LLMs) with external knowledge sources. By combining retrieval and generation, RAG systems can provide more factual, contextually grounded, and up-to-date responses (Shuster et al., 2021; Sahoo et al., 2024). However, the rapid development of RAG architectures (Huang and Huang, 2024; Sarthi et al., 2024), methods (Gao et al., 2021, 2023), and LLM-as-a-Judge metrics (Es et al., 2024) has created a need for an implementation-focused RAG evaluation library to create a reliable way to compare performance on domain-specific datasets.

Assessing a RAG system for a specific dataset requires analyzing the performance of both the retriever and generator, as well as their interaction. It is particularly challenging because LLM outputs can be lengthy, complex, and ambiguous, making it difficult to define clear correctness or objectively measure factual accuracy (Ru et al., 2024). Recent RAG evaluation frameworks, such as RAGAS (Saad-Falcon et al., 2024), RAGChecker (Ru et al., 2024), and TruLens (Snowflake Inc., 2024), have begun addressing these challenges by providing libraries to jointly analyze retrieval and generation quality, reflecting the interdependence between the two components. Instead of relying solely on traditional metrics like BLEU (Papineni et al., 2002) or ROUGE (Lin, 2004), they use LLM-based or embedding-based measures to directly evaluate factual correctness, relevance, and consistency of the generated text (Es et al., 2024). Additionally, they often provide enhanced UI and monitoring features for RAG in production environments.

Despite recent advances, a lack of the following features in state-of-the-art (SOTA) frameworks remains, hindering the effective comparison of RAG methods for domain-specific datasets:

(1) Lack of comparability of scientific methods. Existing libraries in this field focus solely on implementing metrics, with no standardized implementation of SOTA RAG methods. This heterogeneity hinders meaningful and reproducible comparisons across different RAG approaches.

(2) Limited flexibility of LLM-as-a-judge metrics. LLM-as-a-judge metrics are sensitive to domain and dataset, and must be flexible to be effective. Therefore, custom prompts for individually designing evaluations are crucial and are not currently available in existing frameworks.

(3) Primarily cloud-oriented, with limited flexibility for integrating additional metrics or methods. Many existing frameworks are designed with a narrow focus on specific local setups and lack modularity for broader integration. A flexible, plug-and-play architecture is essential to support new RAG methods and additional new metrics.

To tackle these challenges, we present **EncouRAGe**, an open-source Python library for comprehensive, reproducible, and extensible RAG evaluation (Figure 1). It integrates any dataset into a parsable, object-oriented structure for effective and traceable workflows. EncouRAGe includes 10 RAG methods, manages LLM and embedding inference and provides metrics in three categories:

1. **Generator Metrics**: Evaluating the performance end-to-end with classic NLP metrics (ROUGE, BLEU, etc.).

2. **Retrieval Metrics**: Evaluating the effectiveness of the retriever in finding relevant information from the knowledge base.

3. **LLM-as-a-Judge**: Evaluating the effectiveness of the whole RAG Pipeline enhanced by custom LLM prompts.

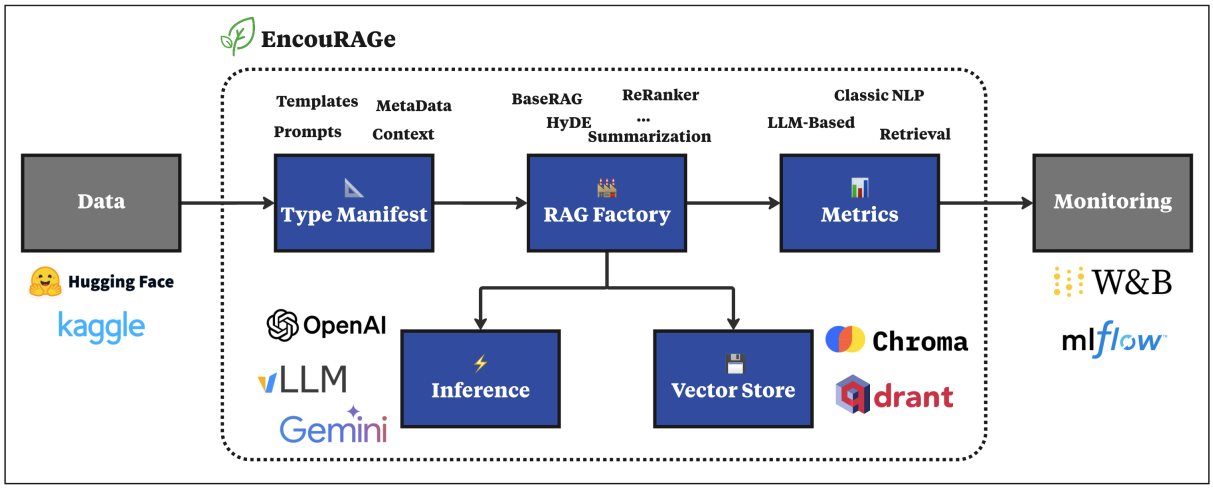

Figure 1: System overview of the **EncouRAGe** Python library. Input data in any format must be transformed to fit the **Type manifest**. The **RAG Factory** provides various RAG methods, while the **Metrics** component implements all evaluation metrics. **Inference** can be executed locally or via cloud providers using the OpenAI SDK, and supported **Vector stores** include Chroma and Qdrant. The output fits popular monitoring systems.

To address the rapid pace and volume of new RAG and LLM research, reproducing and comparing methods has become increasingly time-consuming. With EncouRAGe, evaluating new approaches or verifying others claims becomes straightforward and extensible. The framework enables quick benchmarking across diverse methods and datasets with minimal manual effort, requiring mainly GPU time rather than complex reimplementation. We demonstrate this through experiments comparing multiple RAG methods on four popular QA datasets. The datasets and code are open source and anonymized for submission.[1]
The main contributions of this paper are as follows:

- We present **EncouRAGe**, a novel RAG evaluation framework that consists of **10** SOTA **RAG methods**, an object-oriented **type manifest**, and over **20 metrics** to get transparent and reliable results.
- We conduct a **comparison of RAG methods** using EncouRAGe on four datasets with **25k QA pairs** and **50k documents**, demonstrating the effectiveness and usefulness of the framework.
- We further present one ablation study, showing the effectiveness of **Reranker** ratio for two cross-encoder models and a subset of the four datasets.

[1] https://anonymous.4open.science/r/encourage-B501/

2. Related Work

2.1. Retrieval-Augmented Generation

LLMs excel at text generation but face challenges such as outdated knowledge and hallucinations (Shuster et al., 2021; Huang and Huang, 2024). RAG addresses these issues by incorporating external knowledge, improving accuracy and factuality (Lewis et al., 2020; Sarthi et al., 2024). This approach is critical in domains such as law (Cui et al., 2024), medicine (Xiong et al., 2024; Chen et al., 2025b), and finance (Chen et al., 2025b), where precision is crucial. RAG systems have achieved strong results in tasks such as open-domain question answering (Kim and Lee, 2024), code generation (Wang et al., 2025), and dialogue (Chen et al., 2025a). They have also been successfully adopted in real-world applications, including LlamaIndex (Liu, 2022).

2.2. Evaluation of RAG

Evaluation RAG systems can be categorized into three domains: tools for lightweight and dataset-based evaluation, monitoring frameworks for production applications, and general-purpose evaluation frameworks for LLMs.

(1) Lightweight and Dataset-Based Evaluation Tools. RAGAS (Es et al., 2024) and RAGChecker (Ru et al., 2024) provide systematic metrics for assessing RAG pipelines. RAGAS was the first library to provide LLM-as-a-Judge metrics and offers additional classic NLP evaluation metrics. In contrast, RAGChecker employs a modular approach that decomposes RAG performance into

retrieval and generation components, allowing detailed error attribution and interpretability. But both lack RAG method implementation and are heavily cloud-focused.

(2) Monitoring Frameworks for Production RAG Applications. TruLens (Snowflake Inc., 2024) and Opik (Comet ML, Inc., 2025) extend evaluation capabilities to deployed RAG and LLM-based applications. TruLens enables real-time tracking, custom evaluation functions, and integration with human feedback loops, making it particularly suited for agent-oriented systems where RAG acts as an auxiliary mechanism. Opik provides end-to-end monitoring and analytics for production environments, supporting model observability for improvement workflows. Both systems are application-focused and not designed for research.

(3) General LLM Evaluation Frameworks. DeepEval (AI, 2024) and OpenAI evals (OpenAI, 2025a) offer broader evaluation capabilities that extend beyond RAG-specific scenarios. DeepEval provides standardized testing interfaces and model-agnostic benchmarking tools for assessing reasoning, factual accuracy, and robustness. OpenAI's evals framework enables reproducible and scalable evaluation of LLMs through configurable metrics and task templates.

3. EncouRAGe

As shown in Fig. 1, EncouRAGe is an end-to-end RAG evaluation library in Python consisting of five main components. To sharpen the scope of our framework, we define a task in Subsec. 3.1 and provide an overview of each component to illustrate the framework's idea and workflow. EncouRAGe supports any dataset containing queries, answers, and golden context triples. The data is standardized through the *Type Manifest* as explained in Subsec. 3.2, ensuring type safety and correct assignment of all elements during processing. Users can select from a wide range of RAG methods defined in the *RAG Factory* (Subsec. 3.3), which utilize the *Inference Handler* and *Vector Store* described in Subsec. 3.4. For the main evaluation, EncouRAGe uses over 20 *metrics* across three categories to transparently understand how well RAG methods perform and to identify whether performance loss originates from the retriever or the generator. EncouRAGe features a fully tested codebase and supports all LLMs and embedding models available on Hugging Face and all models implementing the OpenAI SDK.

3.1. Task Formulation

To formally describe the idea and foundation of our framework, which serves as the basis of our library, we define the following mathematical formulations. **EncouRAGe** provides a collection of RAG methods M and evaluation metrics $\mathcal{M}$, which can be applied to any dataset that satisfies the assumptions of $\mathcal{D}$. Let a dataset be defined as

$$\mathcal{D} = \{(q_i, a_i, c_i)\}_{i=1}^{N},$$

where q_i denotes a question, a_i its corresponding answer, and c_i the associated gold context. All contexts c_i are embedded and stored in a vector store

$$\mathcal{V} = \text{Embed}(\{c_i\}_{i=1}^{N}),$$

which serves as the retrieval space for RAG methods. A RAG method is denoted as

$$M = (R, G),$$

where R represents the retriever and G the generator. Given a query q_i, the retriever R accesses the vector store $\mathcal{V}$ using the retrieval strategy defined by M to return a set of top-k relevant contexts

$$C_i = R_k(q_i; \mathcal{V}, M),$$

where $R_k(\cdot)$ denotes the retrieval of the k most relevant contexts from $\mathcal{V}$ according to the similarity function specified by M. C_i is then passed to the generator G to produce a predicted answer

$$\hat{a}_i = G(q_i, C_i).$$

The quality of $\hat{a}_i$ is evaluated against the ground-truth answer a_i using a set of metrics

$$\mathcal{M} = \{m_1, m_2, \ldots, m_K\},$$

yielding performance on multiple metrics

$$s_{i,k} = m_k(\hat{a}_i, a_i), \quad \forall i \in [1, N],\ k \in [1, K].$$

And each metric $s_{i,k}$ helps to understand the performance of the retriever, generator, and their interaction what is the main goal of EncouRAGe.

3.2. Type Manifest

The *Type Manifest* is a collection of object-oriented Python structures designed to streamline evaluation, improve reliability, and enhance maintainability, as illustrated in Fig. 2. Each data point is represented as a `Document` object containing a unique identifier, textual content, and an associated `MetaData` object. The `MetaData` object is a dictionary for auxiliary information not directly used in processing, enabling flexible organization and key-value-based analysis afterward. A `Context` may include multiple `Documents`, reflecting the capability of RAG methods to retrieve several documents from a vector store. Each `Context` is injected into a `Jinja2` template, allowing users to define

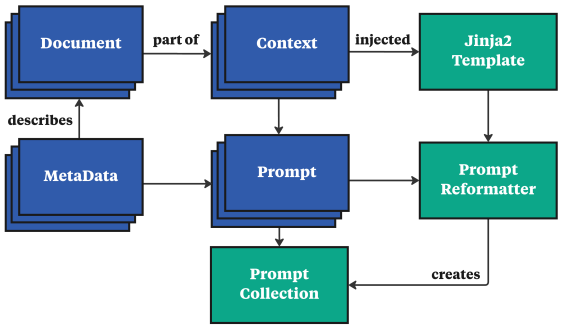

Figure 2: *Type Manifest* of EncouRAGe. Gold **Documents** link to the **Context**, combined via **Jinja2 template** with the prompt to form the final **Prompt Collection. Metadata** ensures traceability for documents and prompts.

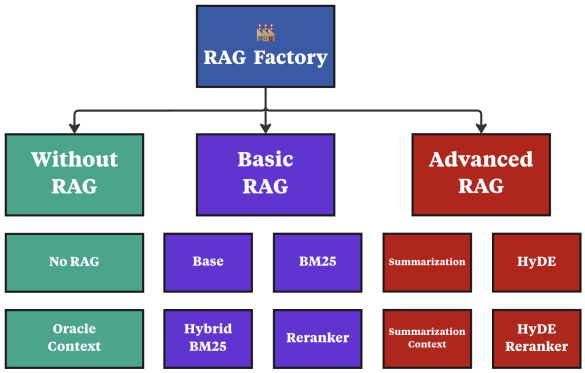

Figure 3: Overview of *RAG Factory* in EncouRAGe. RAG Factory is organized into three categories: Without RAG, Basic RAG, and Advanced. In total, EncouRAGe supports **10 methods**, with more to be added in the future.

prompt structures and dynamically incorporate document content via template variables from `Jinja2`. The `Prompt` object contains an identifier, conversation history (including system and user messages), a `Context`, and a `MetaData` object. When constructing a `PromptCollection`, the `PromptReformatter` renders the `Jinja2` template with the specified system and user prompt, and the related injected template variables. This unified prompt collection ensures consistent usage across the evaluation pipeline, allowing seamless integration with the *RAG Factory* and stable, reproducible metric computation.

3.3. RAG Factory

The *RAG Factory* constitutes the core of our Python library, defining all available methods for dataset evaluation as shown in Fig. 3. Currently, it comprises ten methods grouped into three categories: **Without RAG**, **Basic RAG**, and **Advanced RAG**. The Without RAG category includes methods that assess performance either without any contextual information or with access to golden documents, thereby illustrating the performance gap between having no knowledge and complete knowledge. This comparison indicates the potential performance improvement RAG methods can achieve when the correct context is not known in advance. The distinction between Basic and Advanced RAG lies in the additional step in Advanced RAG, which uses an LLM to generate intermediate outputs. Overall, the library offers a standardized Python interface that facilitates the addition of new methods, while the *Type Manifest* ensures their smooth and consistent integration.

Without RAG. In the *Pretrained-Only* setup, no retriever is employed, and models must answer questions solely based on their pretraining knowledge. Conversely, the *Oracle Context* setting assumes that the relevant context is directly passed to the generator.

Basic RAG Methods. This category includes approaches that retrieve documents using standard embedding-based methods. The *Base RAG* implementation follows the original RAG approach (Lewis et al., 2020), where only the question is embedded to retrieve the top-k documents, which are then passed unchanged to the generator. In addition, a *Standard BM25* (Robertson and Zaragoza, 2009) retriever is provided as a purely lexical baseline, relying on term-frequency and inverse document-frequency weighting to rank documents based on keyword overlap. *Hybrid BM25* (Gao et al., 2021) combines sparse lexical retrieval using BM25 with dense vector retrieval, leveraging both methods to improve recall and relevance. Finally, the *Reranker* method (Tito et al., 2021) can apply any cross-encoder model after initial retrieval to reorder documents based on their significance in a shared embedding space.

Advanced RAG Methods. This category consists of methods that modify the query, transform retrieved contexts, or employ iterative retrieval strategies. The *HyDE* method (Gao et al., 2023) generates hypothetical answers for each question, using them as refined queries to retrieve more relevant documents. *Summarization* reduces noise by summarizing each retrieved context using an LLM, focusing on essential information. *SumContext* applies a similar summarization step, retaining the original full documents for generation, aiming to reduce distractions while preserving content fidelity.

3.4. Inference and Vector Store

The *Inference* and *Vector Store* components are responsible for managing communication with the LLM and the vector database, enabling efficient evaluation of RAG approaches.

For *Inference*, we focus on local deployment, primarily leveraging `vLLM` (Kwon et al., 2023) as it represents one of the fastest LLM engines available. Communication with the LLM is handled through the OpenAI Python SDK (OpenAI, 2025b), allowing seamless integration of HuggingFace, OpenAI, and Gemini. Other integrations are planned.

For the *Vector Store*, we utilize a serverless version of Chroma (Chroma Team, 2025), which creates an in-memory `SQLite3` database stored locally. As an alternative, we also support Qdrant (Qdrant Team, 2025), which is capable of handling large-scale document collections. Both implementations share a unified interface, simplifying the integration of additional vector database backends in the future.

3.5. Metrics

Evaluating different RAG methods is a core feature of our library, implemented through the *Metrics* module. Table 1 gives an overview of the most popular metrics of the library. To provide a transparent and comprehensive assessment, we include metrics for all components of RAG across multiple datasets, grouped into three categories: generator metrics, retrieval metrics, and LLM-based metrics. To align with existing research, we primarily use the `evaluate` (HuggingFace, 2025) library from HuggingFace for most generator metrics. For retrieval metrics, we used the `ir_measures` (MacAvaney et al., 2022) library, which is specialized in information retrieval evaluation. LLM-based metrics are fully implemented and easily customizable with new prompts and examples, enabling flexible use across datasets. The *Type Manifest* typing system is designed to integrate seamlessly with all metrics, enabling automatic computation once a RAG method is selected.

4. Experiments

To demonstrate the effectiveness of our library, we conduct a comprehensive analysis of four popular datasets using each RAG method implemented in EncouRAGe. The datasets used are summarized in Table 2. Each QA pair has at least one associated golden document, enabling comparisons of RAG methods across pretrained, oracle, and various RAG settings. We evaluate a total of nine RAG methods across all datasets, as presented in Table 3, and discuss the results in Subsec. 5.

Metric	Description
Generator Metrics	
Exact Match	Checks if the generated text exactly matches the reference.
Number Match (Anonym)	Checks if generated text is numerical aprox. equal.
BLEU (Papineni et al., 2002)	Measures n-gram overlap between generated and reference text.
ROUGE (Lin, 2004)	Measures overlap of n-grams and sequences.
GLEU (Mutton et al., 2007)	Modified BLEU emphasizing recall and precision balance.
Precision (Sokolova and Lapalme, 2009)	Fraction of generated tokens that are correct.
Recall (Sokolova and Lapalme, 2009)	Fraction of reference tokens correctly generated.
F1 (Sokolova and Lapalme, 2009)	Harmonic mean of precision and recall at token level.
Retrieval Metrics	
Mean Reciprocal Rank (MRR) (Voorhees, 1993)	Mean reciprocal rank of the first relevant document retrieved.
Mean Average Precision (MAP) (Voorhees, 1993)	Mean of average precision scores across queries.
nDCG (Järvelin and Kekäläinen, 2002)	Normalized discounted cumulative gain, emphasizing top-ranked relevance.
Recall@k (Robertson and Zaragoza, 2009)	Fraction of relevant documents retrieved in top-k.
HitRate@k (Li et al., 2019)	Measures if at least one relevant document appears in the top-k results.
LLM-Based Metrics	
Answer Relevance	Assesses if the answer addresses the query.
Answer Faithfulness	Measures factual consistency with the context.
Non-Answer Critic	Detects answers that fail to provide meaningful responses.
Answer Similarity	Compares generated answer to reference or alternative outputs.
Context Recall	Measures how well the retrieved context supports the answer.
Context Precision	Fraction of retrieved context that is relevant to answer.

Table 1: Overview of the most used RAG evaluation metrics that we implemented in *EncouRAGe*.

Subset	Domain	#QA Pairs	Avg. Tokens (Question)	#Documents	Avg. Tokens (Docs)
BioSQA	Biomedical	4,012	13.5	34,634	3,222.8
FinQA	Finance	6,237	45.1	2,110	1,080.4
FeTaQA	Table Questions	7,324	20.4	6,929	520.0
HotPotQA	General Questions	7,395	21.4	7,395	1,421.2
Total	**Multiple**	**24,968**	**25.7**	**51,068**	**2,506.7**

Table 2: Comparison of QA pairs, documents, and average token lengths across subsets. HotPotQA (Yang et al., 2018) & FeTaQA (Nan et al., 2022) are based on Wikipedia articels, while FinQA (Chen et al., 2021) & BioSQA (Krithara et al., 2023) uses PDF documents. Avg. token count based on Gemma-3 tokenizer.

Additionally, we conduct ablation experiments on the reranker models in Subsec. 5.1. We define the reranker ratio as the proportion of initially retrieved documents that are reranked in the second phase, after which a fixed set of 10 papers is provided to the generator model. This should evaluate how different reranker ratios affect BaseRAG's performance and whether combining it with rerankers yields further improvements.

4.1. Experimental Setup

For the evaluation of the dataset, each subset was processed and evaluated independently. First, all contexts were uniquely stored in a vector database using embeddings created with the multilingual e5-large instruct model (Wang et al., 2024). That was done for all RAG methods except for the Summarization, where the summarized context was embedded. A retrieval query was used to retrieve the context from the instruction model. The Top-10 documents were passed to each method of the *RAG Factory* and passed to the *Inference* component in the main evaluation. As generators, we employed Gemma3 27b (Kamath et al., 2025), a SOTA decoder-only transformer. All experiments were conducted on two NVIDIA H100s.

4.2. Evaluation Metrics

We leveraged EncouRAGe's flexibility to evaluate each dataset using task-specific metrics tailored to the required format. For datasets where only a single document contains the gold answer, we employed *MRR@10* to measure the position of the retrieved document. For tasks requiring multiple documents to derive the correct answer, we used *MAP*, in line with prior studies. Additionally, we report *Recall@10* to assess overall retrieval performance, complementing *MRR* by considering the rank of retrieved documents. This is particularly informative in the reranker setup.

For the generation, we also used multiple metrics, depending on the task. For short answers, we used *F1-Score*, for numerical answers, we used *NM* (Anonym).

4.3. Datasets

For evaluation, we selected four datasets from different domains to demonstrate the flexibility of EncouRAGe in handling diverse datasets as presented in Table 2.

HotPotQA. The dataset (Yang et al., 2018) contains 7,395 general knowledge question–answer pairs sourced from Wikipedia, emphasizing multi-hop reasoning. Each question is linked to a single gold document, resulting in 7,395 papers in total, with an average document length of 1,421.2 tokens. Questions average 21.4 tokens, reflecting concise query formulation. HotPotQA challenges models to combine information across multiple articles to answer a single question accurately. For evaluation, we employ *MRR@10* for retrieval and *F1-Score* for answer generation, consistent with multi-hop reasoning tasks.

FetaQA. The dataset (Nan et al., 2022) includes over 7,300 QA pairs that require complex reasoning over tabular data, extracted from Wikipedia articles. The dataset emphasizes semantic understanding and reasoning across multiple table cells. Each question is accompanied by a textual explanation and a gold-standard answer derived from structured tables. We use the training split and all table documents related to these QA pairs, resulting in a total of 6,929 documents to process. Evaluation employs *MRR@10*, *Recall@10*, and *F1-Score* to measure both retrieval effectiveness and answer quality.

FinQA. The dataset (Chen et al., 2021) comprises approximately 6,200 finance-related question–answer pairs derived from company reports and other financial documents. Each question is associated with a single financial document containing a combination of text and tables. On average, there are roughly three QA pairs per document, resulting in a total of 2,110 documents for retrieval. We used a modified version of the training split, reformulating questions to make them more suitable for RAG. For reasons of anonymity, the original pa-

Model	RAG Method	HotPotQA			FeTaQA			FinQA			BioSQA		
		F1	MRR	R@10	F1	MRR	R@10	NM	MRR	R@10	F1	MAP	R@10
Gemma 3 27B + M-E5-Large Instruct	+ Pretrained-Only	36.7	–	–	29.3	–	–	9.6	–	–	39.3	–	–
	+ Oracle Context	43.4	100	100	49.4	100	100	72.9	100	100	54.5	100	100
	+ Base-RAG	37.1	68.8	82.5	49.8	87.5	**92.7**	47.8	45.6	71.9	47.2	42.1	50.1
	+ BM25	40.5	29.1	98.4	39.0	18.6	68.5	50.8	45.5	77.2	44.3	26.7	44.3
	+ Hybrid BM25	**40.8**	70.4	**96.9**	**50.1**	**87.6**	92.4	**51.8**	46.7	**82.1**	**49.9**	43.6	**55.7**
	+ Reranker	36.1	63.4	78.4	49.6	86.9	92.2	50.2	45.5	71.7	44.5	42.1	50.1
	+ HyDE	30.2	40.6	61.0	48.4	82.8	89.5	41.7	37.3	61.4	49.2	**45.9**	54.7
	+ Summarization	31.7	**71.0**	83.3	40.0	83.9	90.5	45.9	**48.5**	74.6	45.0	43.2	51.0
	+ SumContext	37.7	**70.9**	83.3	48.6	84.0	90.5	48.6	48.1	74.6	47.8	43.0	50.4

Table 3: Overall Performance (F1, MRR/MAP, R@10) for Gemma3 with E5-Large Instruct on all four datasets. Cells in **Bold** indicate the highest value over all RAG methods, and underlined indicate the best value across RAG method categories.

per is not cited in this submission. For evaluation, we employ *MRR@10*, *Recall@10*, and *Number Match (NM)*, a metric specifically designed to compare numerical values, like Exact Match, up to the second decimal place.

BioASQ. The dataset (Krithara et al., 2023) contains approximately 4,000 biomedical question–answer pairs derived from scientific literature in the life sciences. It is designed to evaluate factual biomedical question answering in combination with semantic retrieval. The dataset is updated annually, and the version used in our evaluation is BioASQ11. Each QA pair includes multiple reference documents, over 34k in total, making it particularly challenging for the retriever to locate the relevant information. Questions are grouped into four types: list, yes/no, factoid, and summary, each requiring different response formats. For evaluation, we use *MAP* and *Recall@10* for retrieval, and *F1-Score* for generation.

5. Main Results

Table 3 presents the quantitative results across the each of the four QA datasets using each RAG method from Sec. 3.3.

Without RAG Runs. The *Pretrained-Only* results for HotPotQA, FeTaQA, and BioSQA are comparatively high, indicating that these datasets are likely part of the LLM's pretraining corpus (e.g., Wikipedia-based sources). In contrast, FinQA shows a substantially lower score, suggesting that its domain-specific financial questions are less represented in the model's training data. However, when provided with *Oracle Context*, all datasets achieve higher performance than in the *Pretrained-Only*, confirming that the model can effectively reason when supplied with relevant context. Notably, the *Oracle Context* results for HotPotQA (Yang et al., 2018) and FeTaQA (Nan et al., 2022) are slightly below the original papers, as our setup relies solely on a zero-shot setting.

Basic RAG. When incorporating retrieval *Base-RAG* performs close to the *Oracle Context* for HotPotQA, FeTaQA, and BioSQA, while a larger performance gap appears for FinQA. Interestingly, the traditional *BM25* retriever performs comparably to the dense retriever in terms of F1, but exhibits substantially lower MRR on HotPotQA and BioSQA. Combining both retrieval methods (*Hybrid BM25*) yields the most consistent improvements, achieving the highest F1 and Recall@10 across all datasets. The addition of a reranker does not lead to measurable gains, as the MRR remains similar to the base configuration when reordering the top-10 retrieved documents. We discuss the effect of the Reranker further in Subsec. 5.1.

Advanced RAG. Among the advanced retrieval enhancements, *HyDE* offers modest improvements only for BioSQA, which are insufficient to justify its additional computational cost. *Summarization*-based retrieval increases MRR and Recall@10 for all datasets except FeTaQA, but leads to lower F1 scores, likely due to information loss during summarization. The combined approach, *SumContext*, which integrates both summarized and original documents, achieves a balanced trade-off, yielding results similar to *Base-RAG* with slightly better retrieval performance. However, given its additional retrieval and generation overhead, its practical benefit remains limited.

Overall, the results demonstrate that hybrid retrieval remains the most effective and efficient RAG method across diverse QA domains. In contrast, advanced generation-based retrieval strategies offer only marginal improvements at a higher computational cost.

5.1. Reranker Analysis

We further analyzed reranker performance using *EncouRAGe*. A widely discussed claim is that

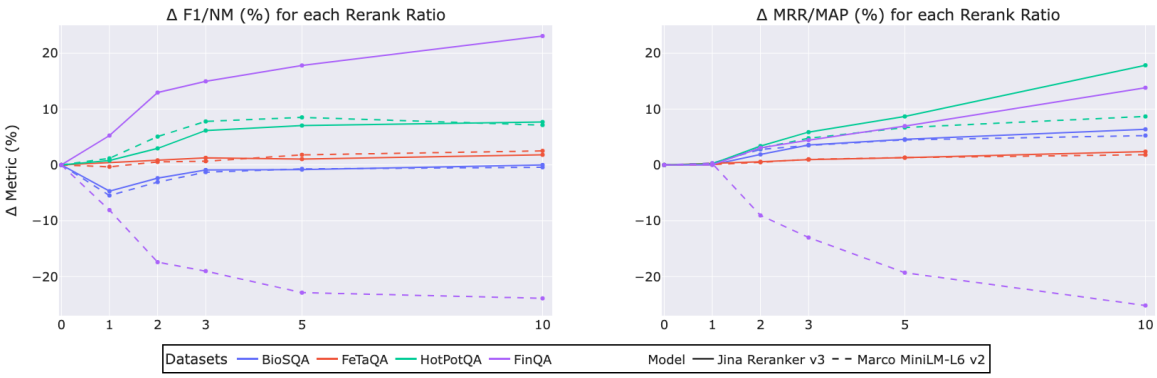

Figure 4: Comparison of **percentage changes for generator (F1/NM) and retrieval metrics (MRR/MAP)** for a 2k-sample subset from each dataset (HotPotQA, FeTAQA, FinQA, and BioSQA). Reranking was performed using Jina v3 and Marco MiniLM-L6 v2. The x-axis represents the reranker ratio, and the y-axis shows the percentage change relative to the base RAG method.

rerankers consistently improve retrieval and generation quality independent of the domain and data format of the documents. Most approaches suggest retrieving a broader set of documents and reranking them before passing the top results to the generator (Clavié, 2024; Bhat et al., 2025).

To investigate this effect, we sampled 2k QA pairs from each of the four datasets and applied two reranker models: Jina V3 (Sturua et al., 2024) and Marco MiniLM-L6 v2 (Reimers and Gurevych, 2020), with five reranker ratios, and compared them against the Base RAG method. For each run, 10 documents were provided to the generator model (Gemma3 27B). A reranker ratio of 1 means only these 10 documents were reranked, while a ratio of 10 indicates that 100 papers were initially retrieved, reranked, and the top 10 were passed to the generator. Evaluation metrics for both generator and retriever followed the same setup as above.

Figure 4 shows the percentage change from the baseline (Base RAG) for generator metrics (F1/NM) and retriever metrics (MRR/MAP) across reranker ratios 1–10. We observe consistent performance gains in the retriever metrics (except for FinQA + Marco), but only meaningful gains in HotPotQA and FinQA. For the generator metrics, even as retrieval performance improved, saturation was observed at a ratio of 3 (except for FinQA + Jina). For FinQA, we hypothesize that the reranker, unlike Jina V3 trained on FinQA samples, was not trained on mixed text–table data, leading to noisy, random rankings.

For FeTaQA, which uses table-based contexts, neither reranker provided notable improvements. This suggests that rerankers offer benefits only when explicitly trained for the data format. Finally, while cross-encoders outperform bi-encoders, reranking increased execution time by 2–4× over the base RAG method, which may pose challenges for production systems that require low-latency responses. Therefore, we recommend using a reranker model only when it is individually trained on the dataset and when high throughput is not the key metric.

6. Conclusion

In this paper, we introduce **EncouRAGe**, a Python framework for evaluating RAG methods locally, reliably, and efficiently. EncouRAGe enables researchers to systematically investigate retrieval-augmented generation techniques on their own datasets, providing insights into which configurations yield the best results for specific domains. The framework includes over 10 RAG methods, more than 20 evaluation metrics, and supports models from Hugging Face, OpenAI, and Gemini. Furthermore, users can flexibly choose between Chroma and Qdrant as the vector store backend.

To demonstrate the capabilities of EncouRAGe, we conducted experiments on four widely used datasets, revealing that the optimal RAG method varies across domains. Overall, our results indicate that the *Hybrid BM25* approach delivers the best performance among the tested configurations. Additionally, we performed an ablation study on reranker models to examine the influence of the reranker ratio. Our findings show that increasing this ratio can boost performance by up to 10%, even if at the cost of higher latency, a trade-off that should be considered when deploying rerankers in real-world systems.

With EncouRAGe, we aim to contribute to the RAG research community by providing a flexible, extensible evaluation framework that enables deeper analysis and supports the development of practical, high-performing systems.

7. Bibliographical References

Confident AI. 2024. DeepEval: The Open-Source LLM Evaluation Framework. Https://deepeval.com.

Riyaz Ahmad Bhat, Jaydeep Sen, Rudra Murthy, and Vignesh P. 2025. UR2N: Unified Retriever and ReraNker. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pages 595–602, Abu Dhabi, UAE. Association for Computational Linguistics.

Yifu Chen, Shengpeng Ji, Haoxiao Wang, Ziqing Wang, Siyu Chen, Jinzheng He, Jin Xu, and Zhou Zhao. 2025a. WavRAG: Audio-Integrated Retrieval Augmented Generation for Spoken Dialogue Models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12505–12523, Vienna, Austria. Association for Computational Linguistics.

Zhe Chen, Yusheng Liao, Shuyang Jiang, Pingjie Wang, YiQiu Guo, Yanfeng Wang, and Yu Wang. 2025b. Towards Omni-RAG: Comprehensive Retrieval-Augmented Generation for Large Language Models in Medical Applications. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15285–15309, Vienna, Austria. Association for Computational Linguistics.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 2021. FinQA: A Dataset of Numerical Reasoning over Financial Data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chroma Team. 2025. Chroma: Open-source search and retrieval database for AI applications. Https://github.com/chroma-core/chroma.

Benjamin Clavié. 2024. rerankers: A Lightweight Python Library to Unify Ranking Methods. ArXiv:2408.17344.

Comet ML, Inc. 2025. Comet — The AI Developer Platform. Https://www.comet.com/site/.

Jiaxi Cui, Munan Ning, Zongjian Li, Bohua Chen, Yang Yan, Hao Li, Bin Ling, Yonghong Tian, and Li Yuan. 2024. Chatlaw: A Multi-Agent Collaborative Legal Assistant with Knowledge Graph Enhanced Mixture-of-Experts Large Language Model. ArXiv:2306.16092.

Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. RAGAs: Automated Evaluation of Retrieval Augmented Generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, St. Julians, Malta. Association for Computational Linguistics.

Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. 2021. Complementing Lexical Retrieval with Semantic Residual Embedding. ArXiv:2004.13969.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise Zero-Shot Dense Retrieval without Relevance Labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.

Yizheng Huang and Jimmy X. Huang. 2024. A Survey on Retrieval-Augmented Text Generation for Large Language Models.

HuggingFace. 2025. Evaluate: A library for easily evaluating machine learning models and datasets. Https://github.com/huggingface/evaluate.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446. Place: New York, NY, USA Publisher: Association for Computing Machinery.

Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, and et al. Mesnard. 2025. Gemma 3 Technical Report. ArXiv:2503.19786.

Kiseung Kim and Jay-Yoon Lee. 2024. RE-RAG: Improving Open-Domain QA Performance and Interpretability with Relevance Estimator in Retrieval-Augmented Generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22149–22161, Miami, Florida, USA. Association for Computational Linguistics.

Anastasia Krithara, James G. Mork, Anastasios Nentidis, and Georgios Paliouras. 2023. The Road From Manual To Automatic Semantic Indexing Of Biomedical Literature: a 10 Years Journey. *Frontiers in Research Metrics and Analytics*, 8.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, pages 611–626, Koblenz, Germany. Association for Computing Machinery.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020*, virtual.

Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Huan Zhao, Pipei Huang, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. 2019. Multi-Interest Network with Dynamic Routing for Recommendation at Tmall. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, pages 2615–2623, Beijing, China. Association for Computing Machinery.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Jerry Liu. 2022. LlamaIndex. Https://github.com/jerryjliu/llama_index.

Sean MacAvaney, Craig Macdonald, and Iadh Ounis. 2022. Streamlining Evaluation with ir-measures. In *Advances in Information Retrieval*, pages 305–310, Cham. Springer International Publishing.

Andrew Mutton, Mark Dras, Stephen Wan, and Robert Dale. 2007. GLEU: Automatic Evaluation of Sentence-Level Fluency. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 344–351, Prague, Czech Republic. Association for Computational Linguistics.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, Dragomir Radev, and Dragomir Radev. 2022. FeTaQA: Free-form Table Question Answering. *Transactions of the Association for Computational Linguistics*, 10:35–49.

Irina Nikishina, Özge Sevgili, Mahei Manhai Li, Chris Biemann, and Martin Semmann. 2025. Creating a Taxonomy for Retrieval Augmented Generation Applications. ArXiv:2408.02854.

OpenAI. 2025a. OpenAI Evals — A framework for evaluating LLMs. Https://github.com/openai/evals.

OpenAI. 2025b. openai-python: Official Python library for the OpenAI API. Https://github.com/openai/openai-python.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Qdrant Team. 2025. Qdrant: High-performance, massive-scale Vector Database and Vector Search Engine for the next generation of AI. Https://github.com/qdrant/qdrant.

Nils Reimers and Iryna Gurevych. 2020. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Dongyu Ru, Lin Qiu, Xiangkun Hu, Tianhang Zhang, Peng Shi, Shuaichen Chang, Cheng Jiayang, Cunxiang Wang, Shichao Sun, Huanyu Li, Zizhao Zhang, Binjie Wang, Jiarong Jiang, Tong He, Zhiguo Wang, Pengfei Liu, Yue Zhang, and Zheng Zhang. 2024. RAGChecker: A Fine-grained Framework for Diagnosing Retrieval-Augmented Generation. In *Advances in Neural Information Processing Systems*, volume 37, pages 21999–22027. Curran Associates, Inc.

Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. 2024. ARES: An Automated Evaluation Framework for Retrieval-Augmented Generation Systems. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 338–354, Mexico City, Mexico. Association for Computational Linguistics.

Pranab Sahoo, Prabhash Meharia, Akash Ghosh, Sriparna Saha, Vinija Jain, and Aman Chadha.

2024. A Comprehensive Survey of Hallucination in Large Language, Image, Video and Audio Foundation Models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11709–11724, Miami, Florida, USA. Association for Computational Linguistics.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. In *The Twelfth International Conference on Learning Representations*, Vienna, Austria. The Association for Computational Linguistics.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval Augmentation Reduces Hallucination in Conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Snowflake Inc. 2024. TruLens Eval. Https://www.trulens.org.

Marina Sokolova and Guy Lapalme. 2009. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437.

Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Nan Wang, and Han Xiao. 2024. jina-embeddings-v3: Multilingual Embeddings With Task LoRA. ArXiv:2409.10173.

Rubèn Tito, Dimosthenis Karatzas, and Ernest Valveny. 2021. Document Collection Visual Question Answering. In *16th International Conference on Document Analysis and Recognition*, volume 12822 of *Lecture Notes in Computer Science*, pages 778–792, Lausanne, Switzerland. Springer.

Ellen M. Voorhees. 1993. Using WordNet to disambiguate word senses for text retrieval. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '93, pages 171–180, Pittsburgh, Pennsylvania, USA. Association for Computing Machinery.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual E5 Text Embeddings: A Technical Report. ArXiv:2402.05672.

Yuchen Wang, Shangxin Guo, and Chee Wei Tan. 2025. From Code Generation to Software Testing: AI Copilot With Context-Based Retrieval-Augmented Generation. *IEEE Software*, 42(4):34–42.

Guangzhi Xiong, Qiao Jin, Zhiyong Lu, and Aidong Zhang. 2024. Benchmarking Retrieval-Augmented Generation for Medicine. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6233–6251, Bangkok, Thailand. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.