BES4RAG: A Framework for Embedding Model Selection in Retrieval-Augmented Generation

Lorenzo Canale[1,*,†], Stefano Scotta[1,*,†], Alberto Messina[1,*] and Laura Farinetti[2]

[1]*RAI - Centro Ricerche, Innovazione Tecnologica e Sperimentazione, Via Giovanni Carlo Cavalli 6, 10138, Turin, Italy*
[2]*Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129, Turin, Italy*

Abstract
Embedding model selection is a crucial step in optimizing Retrieval-Augmented Generation (RAG) systems. In this paper, we introduce **BES4RAG**, a framework designed to evaluate embedding models based on question-answering accuracy rather than standard retrieval metrics. BES4RAG automates dataset processing, automatic question generation, passage indexing, retrieval, and answer evaluation to determine the optimal embedding model for specific datasets. Experimental results on three diverse datasets confirm that embedding choice significantly affects performance, varies across datasets, and can enable smaller LLMs to outperform larger ones when paired with the right embeddings. Additionally, since a key component of this framework is automatic question generation, we found that its performance closely aligns with manually crafted questions, as evidenced by the Pearson correlation between the two.

Keywords
Embedding Model Selection, Automatic Question Generation, Evaluation Framework, Retrieval-Augmented Generation (RAG),

1. Introduction

Retrieval-Augmented Generation (RAG) has emerged as a powerful approach for improving the factual accuracy and contextual relevance of Large Language Models (LLMs) by incorporating external knowledge sources [1]. A crucial component of a RAG system is the embedding model, which converts textual data into vector representations for retrieval [2, 3, 4, 5]. Standard retrieval metrics like Recall@k, Mean Reciprocal Rank (MRR), Normalized Discounted Cumulative Gain (NDCG), Mean Average Precision (MAP), and Precision at some cutoff (Precision@k) are commonly used to evaluate embeddings [6], but they do not always reflect how well retrieved passages enhance answer quality. Additionally, these metrics require knowing the source document of key answer components, yet this information is not always easily accessible.

In this work, we introduce **BES4RAG**, a framework designed to address these limitations by focusing on evaluating embedding models based on their impact on question-answering accuracy, rather than relying solely on traditional retrieval metrics.

BES4RAG implements a fully automated pipeline that processes datasets, generates multiple-choice questions (MCQs) using an LLM, indexes passages using different embedding models, retrieves relevant documents, and evaluates the accuracy of generated answers. By comparing retrieval-augmented responses across different embeddings and LLM configurations, BES4RAG enables practitioners to identify the best embedding model for their specific dataset and use case.

We used BES4RAG to conduct a series of experiments on three diverse types of datasets: news articles, TV program transcripts, and movie-related data — including both scripts and additional metadata — each with varying lengths and characteristics, addressing three key research questions.

RQ1 **Are optimal embedding choices dataset-dependent?** We demonstrate that different datasets yield significantly different optimal embeddings, reinforcing the importance of dataset-specific selection.

RQ2 **Can small LLMs outperform larger models when paired with the right embeddings?** Our findings suggest that embedding quality can play a more significant role than LLM size, highlighting the necessity of embedding optimization.

RQ3 **Do results from automatically generated questions correlate with those from manually created ones?** We validate that automated question evaluation is a reliable proxy for human-generated assessments, confirming the robustness of BES4RAG's methodology.

CLiC-it 2025: Eleventh Italian Conference on Computational Linguistics, September 24 — 26, 2025, Cagliari, Italy
*Corresponding author.
†These authors contributed equally.

lorenzo.canale@rai.it (L. Canale); stefano.scotta@rai.it (S. Scotta); alberto.messina@rai.it (A. Messina); laura.farinetti@polito.it (L. Farinetti)
0000-0002-7556-595X (L. Canale); 0000-0003-1078-2985 (S. Scotta); 0000-0002-8262-2449 (A. Messina); 0000-0001-8614-4192 (L. Farinetti)

In summary, our results emphasize the importance of evaluating embedding models based on their impact on question-answering accuracy, with a methodology that minimizes user effort through the automatic generation of questions.

2. Related Work

The Massive Text Embedding Benchmark (MTEB) provides a valuable overview of the performance of hundreds of embedding models across a variety of tasks and datasets [7]. However, it also presents some limitations. Even when models are evaluated on multiple datasets for a given task, these datasets rarely match the specific characteristics — such as language, document length, or corpus size — of the data a user might use to build a RAG system. Additionally, for retrieval tasks, the evaluation metrics adopted by MTEB may not be fully appropriate in scenarios where the same information is spread across multiple documents. In such cases, the ranking of individual documents becomes less meaningful, as the relevant information is redundantly present in several of them.

For these reasons, *new evaluation methods are emerging in the literature that incorporate Large Language Models (LLMs)* [8]. For example, in [9], the capabilities of ChatGPT and Llama2 are leveraged to evaluate embedding models in the context of RAG. Instead of relying solely on retrieval metrics, ChatGPT is used to rank the relevance and usefulness of the context retrieved by different embedding models. In [10], the authors propose a clustering-based approach to analyze the behavior of *embedding models within RAG systems*. By grouping models into families based on their retrieval characteristics, the study reveals that top-k retrieval similarity can show high variance across different model families, especially at lower values of k. This highlights how seemingly similar models may behave quite differently in practice, reinforcing the importance of dataset-specific and task-aware embedding evaluation. More recent work has further emphasized the importance of considering embedding performance specifically within RAG pipelines. Sakar and Emekci, in [11], show that balancing context quality with similarity-based ranking is crucial, along with understanding trade-offs related to token usage, runtime, and hardware constraints. Their findings highlight the role of contextual compression filters in improving hardware efficiency and reducing token consumption, despite their effect on similarity scores. Similarly, in [12] COCOM is introduced, a context compression method that reduces long input contexts to a small set of compact embeddings. This approach significantly accelerates generation time by mitigating the overhead introduced by lengthy contextual inputs, which directly impacts user latency.

In parallel, the *automatic generation of questions using LLMs* has gained attention, especially in educational and evaluation contexts. In [13] it is presented a system that allows users to specify a question type (e.g., reading, speaking, or listening) and a base text, from which the system automatically generates questions accordingly. A more structured approach with PFQS (Planning First, Question Second) is proposed in [14], in which Llama 2 generates an answer plan that is then used to produce relevant questions. While these methods demonstrate the potential of LLMs for generating educational content, the systematic use of automatically generated questions for evaluating embedding performance in RAG systems remains underexplored and merits further investigation.

3. BES4RAG: A Framework for Selecting Embeddings in RAG.

BES4RAG (Benchmarking Embeddings for Selection in RAG) is a modular framework written in Python code and designed to assess embedding models *end-to-end* by evaluating their performance in the full RAG pipeline, rather than relying solely on pre-retrieval metrics.

BES4RAG differs from conventional evaluation methods by integrating automated question generation and response evaluation within the RAG loop. This enables a direct comparison of how different embeddings affect the final output quality, making the framework suitable for real-world, task-specific deployment.

The framework, depicted in Figure 1, is publicly available on GitHub.[1] In the following sections, we describe the individual pipeline modules.

3.1. Data Preprocessing: File Conversion and Organization

The preprocessing phase is handled by a module that ingests a variety of input formats—namely JSON, TXT, and PDF files—and converts them into plain text for downstream processing. This module also creates a `file_mapping.json` file, which records the correspondence between the original input and the resulting text files. Optionally, a brief textual description can be associated with each input document. This description can be generated automatically based on the original filename or derived from the content using a large language model (LLM); alternatively, the user can manually specify it. This step ensures that the dataset is normalized, forming the foundation for consistent question generation and passage segmentation in later stages.

[1]https://github.com/RaiCRITS/BES4RAG

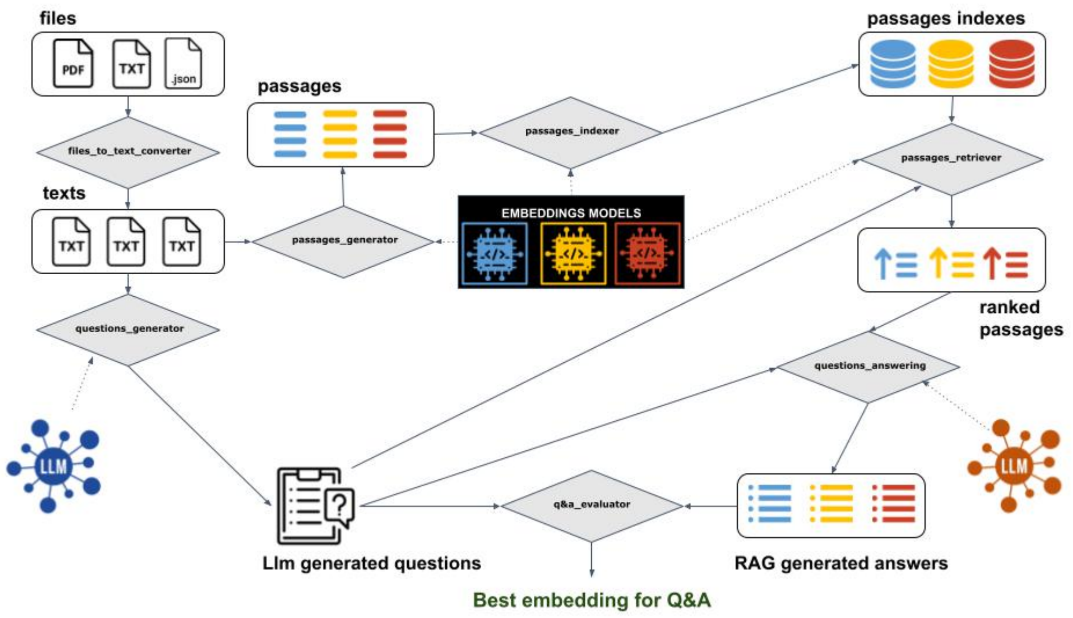


Figure 1: BES4RAG framework pipeline. The schema shows the whole pipeline starting (top left) from the documents which are converted into text files and then split into passages, see Section 3.1. The embedding of the passages are then computed and stored in indexes (top right), see Section 3.4. The text files are then sampled and given to an LLM appropriately prompted to automatically generate multiple-choice questions (bottom left), see Section 3.2. For each of these questions and for each embedding model, we rank the passages based on their similarity to the question (center right), see Section 3.5. The generated questions are then prompted to an LLM with the top k retrieved passages for all the embedding models and different values of k, collecting the answers given (bottom right), see Section 3.6. Lastly, comparing the answers given by the LLM in the previous phase with the correct ones, generated alongside the questions, it is possible to evaluate which embedding model performs best for the particular dataset considered (bottom), see Section 3.7.

3.2. Automatic Questions Generation

A central component of BES4RAG is the automatic generation of MCQs from the input text. Using a LLM, the `questions_generator` module selects random text segments from the normalized dataset and formulates MCQs based on a customizable prompt template. The standard prompt used for question generation is in Figure 2. The questions are stored in JSON format.

3.3. Text Segmentation

Once the dataset is converted into text files, it is segmented into passages suitable for indexing. The `passages_generator` module performs this task by applying a specified tokenizer to the input text. A key consideration in this process is that the segmentation into passages is determined by the embedding model being used since the tokenizers have a maximum token length. By default, the framework uses the maximum token length supported. However, it is possible to specify a smaller token length.

3.4. Passages Indexing

The segmented passages are embedded using one or more embedding models via the `indexer` module. This module computes and stores vector representations of the passages.

3.5. Passages Retrieval

Given a set of questions and indexed embeddings, the `passages_retriever` module ranks the passages based on similarity, typically using cosine similarity, though other similarity metrics can be employed depending on the embedding model. The retrieved passages are then stored, organized by embedding model, allowing for flexible experimentation with different top-k retrieval sizes.

3.6. Question Answering

Using the retrieved passages and corresponding questions, the `questions_answering` module evaluates how well an LLM can answer each question in a RAG

```
Create a multiple-choice question in the same language
as the text below, based solely on its content.

---------------------------
<<<text>>>
---------------------------

The question must be generic and must not contain
references to the article (e.g., "in the article..." or
 "based on the text").

If the text mentions a specific event, include full
details (e.g., name of war, date if available). Avoid
vague temporal references like "today."

Generate 4 answer options (1 correct, 3 plausible but
incorrect), each with an explanation of why it is
correct or not, based only on the text.

Return your answer in this JSON format:

{
  "question": "...",
  "options": [
    {
        "text": "...",
        "is_correct": true/false,
        "explanation": "..."
    },
    ...
  ]
}

Return only the JSON object in the same language as the
 input.
```

Figure 2: Prompt used for automatic question generation.

setup. For each value of k (with default values of $k = 0, 1, 2, 3, 4, 5, 10$), the module combines the top-k retrieved passages with the question prompt and queries an LLM to generate an answer. The prompt used for let the LLM answer the questions is in Figure 3. The results are stored in structured JSON files, organized by embedding and LLM configuration.

```
Answer the following multiple-choice question:

<<<multiple choice question>>>

using the following textual documents as possible
sources:

*****
<<<k passages retrieved>>>
*****

Respond by providing only the numerical identifier of
the correct answer from the options 0, 1, 2, 3.
Do not respond with anything other than one of these
numbers even if you do not know the answer.
```

Figure 3: Prompt used for question answering.

3.7. Evaluation

The final module, `q&a_evaluator`, assesses the performance of the RAG system across different embeddings by computing the answer accuracy over all questions. For each embedding model and retrieval configuration (e.g., varying k), the module calculates accuracy and generates a plot to visualize performance. This plot is crucial for identifying the embedding model that leads to the best overall performance in the specific domain or dataset under analysis. Additionally, it helps determine the optimal value of k for the considered task. This evaluation also enables a comparison between free and open-source embedding models and their proprietary counterparts, providing insights into the trade-offs between computational cost and accuracy.

4. Experimental Setup

In this section, we describe the experimental setup used to evaluate the performance of the proposed system. We first provide an overview of the datasets used, followed by details about the embedding models and LLMs employed in the pipeline. Finally, we explain the evaluation metric adopted to measure the system's performance in answering questions.

4.1. Datasets

We evaluate our system on three distinct datasets, each representing a different domain and content type. These datasets were selected to test the system's versatility and ability to generalize across varying text types, from news articles to transcripts of TV programs and movie scripts.

- **RaiNews:** This dataset consists of approximately 16,000 news articles, from the RaiNews portal, covering a wide range of topics from current events. The articles are typically short and serve as concise textual documents, ideal for testing the system's ability to retrieve and generate answers from concise content.
- **Medicina33:** This dataset includes roughly 159 full transcripts from the *Medicina 33* TV program. This Italian television program focuses on medical topics, with discussions featuring experts in the field of medicine. The transcripts are longer with respect to the news, making them suitable for testing the system's handling of more complex, specialized content.
- **Movies:** This dataset comprises approximately 2,000 movie scripts, metadata, and reviews. It includes both short and long documents, providing a diverse set of examples ranging from concise

Table 1
Embedding models adopted for all three datasets

Type	Model
ColBERT	antoinelouis/colbert-xm
openai	text-embedding-3-large
openai	text-embedding-3-large (512 token limit)
Sentence Transformers	intfloat/multilingual-e5-large
Sentence Transformers	sentence-transformers/all-MiniLM-L6-v2
Sentence Transformers	dunzhanq/stella_en_1.5B_v5

summaries to lengthy dialogues. This dataset is intended to evaluate the system's performance on text with a narrative structure and its ability to handle various types of content, such as reviews and scripts.

The RaiNews and Medicina33 datasets are in Italian, while the Movies dataset is in English.

4.2. Embedding Models

In our experiments, we distinguish between three main families of embedding models: ColBERT, OpenAI embeddings, and Sentence Transformers.

The *ColBERT* model, described in [15], is a state-of-the-art method for efficient and effective passage retrieval. ColBERT uses a bi-level representation of text, allowing for a more compact and computationally efficient representation of passages. The antoinelouis/colbert-xm[2] model, based on this framework, is a multilingual variant, providing advantages in multilingual tasks by capturing semantic meaning in multiple languages simultaneously.

Openai offers a range of powerful models for generating embeddings from text, including the text-embedding-3-large[3] model. The main disadvantage of these models is that they are proprietary, and the vector representation is available only through a paid API.

The *Sentence Transformers* family includes several models optimized for sentence-level embeddings.

- intfloat/multilingual-e5-large[4][16]: A multilingual model capable of generating high-quality embeddings for text in multiple languages.
- sentence-transformers/all-MiniLM-L6-v2[5] [17]: A smaller, faster variant of the BERT model, providing efficient sentence embeddings while maintaining a high degree of accuracy for various NLP tasks.
- dunzhanq/stella_en_1.5B_v5[6] [18]: A large-scale transformer model fine-tuned for English sentence-level tasks, designed to provide powerful embeddings for more complex textual data.

Remark 1. We selected primarily multilingual embedding models since our experiment involves two datasets in Italian and one in English (see Section 4.1), to reduce potential mismatches between dataset languages and model training data. This choice ensures broader language coverage and more robust cross-lingual representations. However, BES4RAG does not aim to recommend a specific model a priori, but rather to evaluate a user-defined set of models and identify the best-performing one for the dataset considered.

To compare the embeddings produced by these models, the most common *similarity measure* is cosine similarity, which computes the cosine of the angle between two vectors, capturing their relative orientation in the embedding space. Cosine similarity is used for all models in our setup except for those in the *ColBERT* family. For the latter, such as antoinelouis/colbert-xm, we instead use the *MaxSim* function, a more specialized similarity measure designed for passage retrieval that works by first computing the similarity between each individual query token and each document token using a similarity metric like cosine similarity; it then takes the maximum of these token-level similarities as the final relevance score between the query and the document.

Finally, for all datasets, the maximum token limits for embeddings were applied to split the textual data into passages, except for the OpenAI model text-embedding-3-large (512 token limit), which is the same model as text-embedding-3-large but with maximum tokens length limited to 512. The decision of considering also this case was made based on the observation that increasing the size of passages, although possible with this model, does not necessarily improve the quality of the retrieved information. This will become clear when observing the results in Section 5.

[2]https://huggingface.co/antoinelouis/colbert-xm
[3]https://platform.openai.com/docs/models/text-embedding-3-large
[4]https://huggingface.co/intfloat/multilingual-e5-large
[5]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2
[6]https://huggingface.co/dunzhanq/stella_en_1.5B_v5

4.3. Large Language Models

In our experimental setup, we employed two distinct families of LLMs for the generation of questions and answering, respectively. For question generation, `GPT-4o`[7] model was adopted through the OpenAI API. For answering, we adopted two variants of the LLaMA 3.1 series developed by Meta: the 70-billion parameter model `meta-llama/Llama-3.1-70B-Instruct`[8] and the smaller 8-billion parameter version `meta-llama/Llama-3.1-8B-Instruct`[9].

To ensure consistency and reduce stochastic variation across outputs, a temperature of 0 was used during inference for all models. Additionally, for answer generation tasks, the maximum output length was restricted to a single token, since the expected answer is always a discrete value in the set $\{0, 1, 2, 3\}$, in accordance with the prompt specification described in Section 3.6.

4.4. Evaluation Metric

Unlike to what is done in [9], we do not aim to evaluate the performance of our embedding models using a LLM as an external judge. In other words, we do not rely on the LLM to assess the quality of the retrieved passages or to rate their relevance. Instead, we consider the end goal of the pipeline: whether the final multiple-choice answer produced by the RAG system is correct.

To this end, we introduce a simple yet informative metric that we refer to as *Question Answering Accuracy* — or simply *accuracy* in the remainder of this paper. For each question, the system selects an answer option based on the response generated by the LLM, using the passages retrieved by the embedding model. The accuracy is computed as the proportion of questions for which the selected answer matches the correct one, as defined in the ground truth. This metric directly reflects the effectiveness of the entire RAG pipeline in producing correct answers, integrating both retrieval and generation performance.

Remark 2. Theoretically, the pipeline could be adapted to incorporate standard retrieval metrics such as those mentioned in Section 1, by changing the question generation module so that questions are generated from individual passages rather than from full documents. However, we adopt the *Question Answering Accuracy* metric for its direct alignment with the end goal of the RAG pipeline: selecting the embedding that enables correct answers. While we acknowledge its binary nature and the lack of granularity in capturing partial understanding or passage quality, we consider this trade-off acceptable for an automated evaluation setup. More expressive metrics often require detailed annotations that are not always available.

[7]https://openai.com/index/hello-gpt-4o/
[8]https://huggingface.co/meta-llama/Llama-3.1-70B-Instruct
[9]https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct

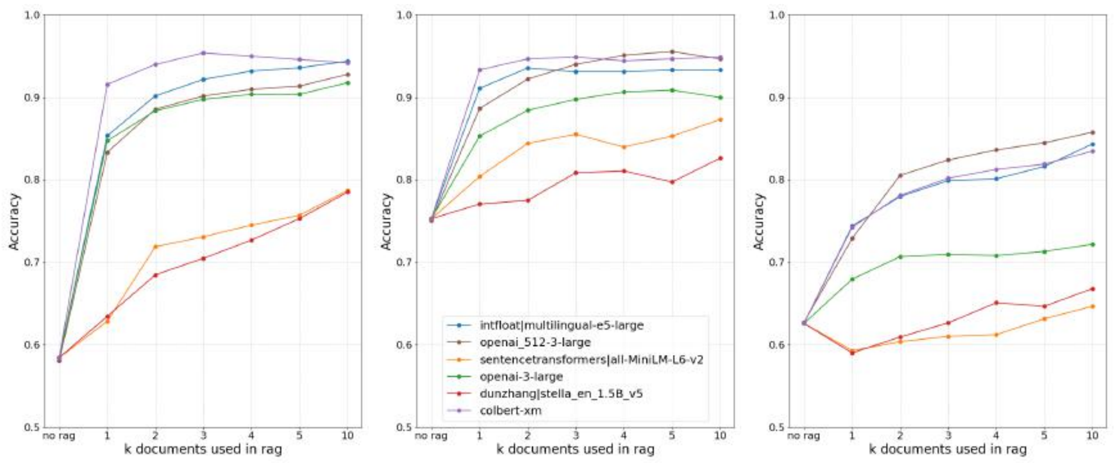
5. Results and Discussion

RQ1: Optimal embedding choices vary across datasets

As observed in Figure 4, the accuracy of the `Llama 3.1 70B` model on automatically generated questions exhibits variations not only with the number of retrieved documents, but also with respect to the choice of embedding model. The ranking of the embedding models varies across datasets, as demonstrated by the different performance patterns observed in the first and subsequent positions. This variation highlights the dataset-specific characteristics that influence the efficacy of embedding models, further emphasizing the utility of the proposed framework for selecting the optimal embeddings for each dataset, rather than relying on a one-size-fits-all approach.

RQ2: Small LLMs can outperform bigger LLMs with the right embedding

In some cases, the choice of the embedding model may be even more critical than selecting the most powerful LLM within a RAG system. This hypothesis is supported by experimenting BES4RAG using two different LLMs framework on the same dataset and with the same embedding models. As shown in Figure 5, these experiments demonstrate that using a more effective embedding model with a smaller LLM can lead to better performance than relying on a more powerful LLM combined with weaker embedding models. In particular, LLama 3.1 8B, when paired with antoinelouis/colbert-xm, intfloat/multilingual-e5-large, or text-embedding-3-large, outperforms the larger LLama 3.1 70B when the latter is combined with sentence-transformers/all-MiniLM-L6-v2 or dunzhanq/stella_en_1.5B_v5, at least for lower values of k. Indeed, for higher values of k, the performance of the smaller LLM deteriorates, likely due to the increased prompt length exceeding its optimal processing capacity. These experiments highlight the importance of carefully evaluating the choice of the embedding model, especially when considering the use of smaller LLMs. In fact, selecting an effective embedding model can enable the adoption of smaller language models, thus reducing computational requirements and leading to more cost-effective and resource-efficient solutions.



(a) RaiNews (b) Medicina33 (c) Movies

Figure 4: Accuracy comparison of LLama 3.1 70B on automatically generated questions from different datasets depending on the embedding models and the number of retrieved documents used to answer the questions (x-axis). The legend (the same for the three plots), with shortened names for the models in Section 4.2, is displayed only in (b) to not compromise the readability of the plots.

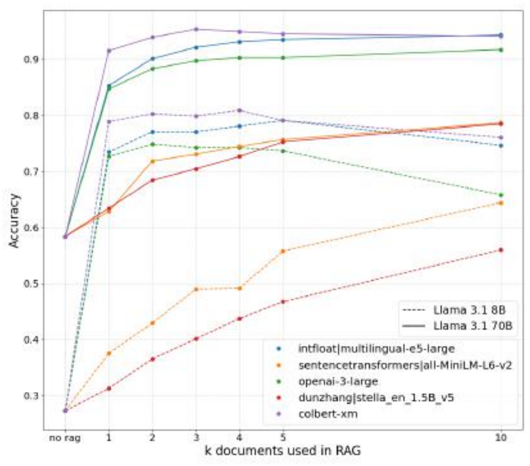


Figure 5: Accuracy comparison between Llama 3.1 8B and Llama 3.1 70B on automatically generated questions from the Rainews dataset depending on the embedding models (the ones in Section 4.2, here with shortened names) and the number of retrieved documents used to answer the questions (x-axis).

RQ3: Automatically generated and user-generated questions

To assess whether evaluation using automatically generated questions provides results consistent with human-authored ones, we relied on a manually curated set of 1,414 questions created by approximately eighty students enrolled in an undergraduate database course. These students were instructed to formulate meaningful and unambiguous multiple-choice questions based on the movies scripts, plots and metadata.

Table 2
Pearson correlation between accuracy matrices obtained from manual and automatic questions for the *Movies* dataset, using different normalization strategies.

Normalization Strategy	Pearson Correlation (r)
None (raw scores)	0.78
Min-max per row	0.80
Min-max over full matrix	0.90

We then compared the accuracy scores obtained using these human-authored questions with the automatically generated ones for the *Movies* dataset. Specifically, for each embedding model and for each value of k in the top-k retrieval, we computed the accuracy of the final answers returned by the RAG pipeline. This yielded two matrices of scores: one for manual questions and one for automatically generated questions, where rows correspond to different embedding models and columns to different k values.

We then calculated the Pearson correlation coefficient between the corresponding entries of these two matrices to quantify the alignment between the two evaluation modes. As shown in Table 2, the raw accuracy values already exhibit a strong correlation ($r = 0.78$). When applying min-max normalization per row (i.e., within each embedding), the correlation improves slightly ($r = 0.80$), indicating that the relative behavior of each model across different k remains consistent. Finally, full matrix-wise normalization further increases the correlation to $r = 0.90$, suggesting a strong structural similarity between the two evaluation matrices. These findings support the use of automatically generated questions as a viable proxy for manual evaluation.

Remark 3. In addition to the quantitative correlation analysis, we manually inspected a random sample of both human and automatically generated questions to assess their coherence and correctness. The review confirmed a high level of quality in both sets. The automatically generated questions typically referred to more specific and localized portions of the source text. Anyway, the strong correlation observed between the two evaluation modes further supports the use of automatically generated questions as a reliable and efficient benchmark for assessing embedding model performance.

6. Conclusion and Future Work

In this work, we presented BES4RAG, a modular framework for the evaluation of embedding models in retrieval-augmented generation (RAG) pipelines. The framework provides a comprehensive approach by focusing on end-to-end evaluation, incorporating automatic question generation, passage segmentation, and answer evaluation. Unlike traditional methods, which rely on pre-retrieval metrics, BES4RAG integrates task-specific performance assessments, allowing for a more accurate comparison of embedding models based on their impact on the final output.

BES4RAG is also versatile, making it suitable for a variety of use cases, including datasets that represent subsets of larger corpora. A prime example would be transcribed multimedia archives, where smaller portions of the dataset can be used to effectively represent the entire collection.

Although BES4RAG demonstrates strong performance and general applicability across diverse datasets, it is not without limitations. One notable limit lies in its reliance on automatically generated MCQs, which, although efficient and scalable, may not always be adequate in highly domain-specific contexts, i.e. in technical or expert-driven fields where factual precision or nuanced phrasing is critical. Furthermore, the binary nature of the evaluation metric is easily interpretable, but it can fail to capture partial understanding, near-miss responses, or the contextual relevance of the retrieved passages. This trade-off between simplicity and expressiveness, while intentional for automation and reproducibility, highlights the need for complementary metrics or qualitative assessments in more complex scenarios.

Looking ahead, avenues for future work include the following:

- Investigating whether using two different LLMs for question generation and retrieval provides better performance or if using the same LLM for both tasks yields comparable results.
- Exploring alternative methods for question generation that consider larger portions of documents.
- Introducing new metrics to assess questions without options, potentially linking detailed answers back to one of the predefined options, offering more flexibility in evaluating the question-answer generation process.
- Integrate within the pipeline some element that returns statistical significance measures of the results obtained, such as paired tests to assess whether differences between embedding models are statistically significant. Moreover, regarding the evaluation of LLM's answers it could be interesting to analyze the token-level probability distribution to assess how embeddings affect the confidence of LLM predictions.
- Study the scalability of the proposed approach on significantly larger datasets, evaluating both its performance and reliability under increased data volume, as well as the computational time and resource requirements of the entire pipeline.

References

[1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive nlp tasks, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20, Curran Associates Inc., Red Hook, NY, USA, 2020, pp. 9459–9474.

[2] R. Egger, Text Representations and Word Embeddings, Springer International Publishing, Cham, 2022, pp. 335–361. URL: https://doi.org/10.1007/978-3-030-88389-8_16. doi:10.1007/978-3-030-88389-8_16.

[3] T. Kim, J. Springer, A. Raghunathan, M. Sap, Mitigating bias in rag: Controlling the embedder, 2025. URL: https://arxiv.org/abs/2502.17390. arXiv:2502.17390.

[4] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2019. URL: https://arxiv.org/abs/1908.10084.

[5] S. Wang, R. Koopman, Semantic embedding for information retrieval, in: 5th Workshop on Bibliometric-Enhanced Information Retrieval, BIR 2017, CEUR, 2017, pp. 122–132.

[6] F. Radlinski, N. Craswell, Comparing the sensitivity of information retrieval metrics, in: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '10, Association for Computing Machinery, New York, NY, USA, 2010, p. 667–674. URL: https://doi.org/10.1145/1835449.1835560. doi:10.1145/1835449.1835560.

[7] N. Muennighoff, N. Tazi, L. Magne, N. Reimers, MTEB: Massive text embedding benchmark, in: A. Vlachos, I. Augenstein (Eds.), Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, Dubrovnik, Croatia, 2023, pp. 2014–2037. URL: https://aclanthology.org/2023.eacl-main.148/. doi:10.18653/v1/2023.eacl-main.148.

[8] J. Isbarov, K. Huseynova, Enhanced document retrieval with topic embeddings, in: 2024 IEEE 18th International Conference on Application of Information and Communication Technologies (AICT), 2024, pp. 1–5. doi:10.1109/AICT61888.2024.10740455.

[9] S. Kukreja, T. Kumar, V. Bharate, A. Purohit, A. Dasgupta, D. Guha, Performance evaluation of vector embeddings with retrieval-augmented generation, in: 2024 9th International Conference on Computer and Communication Systems (ICCCS), 2024, pp. 333–340. doi:10.1109/ICCCS61882.2024.10603291.

[10] L. Caspari, K. G. Dastidar, S. Zerhoudi, J. Mitrovic, M. Granitzer, Beyond benchmarks: Evaluating embedding model similarity for retrieval augmented generation systems, 2024. URL: https://arxiv.org/abs/2407.08275. arXiv:2407.08275.

[11] T. Şakar, H. Emekci, Maximizing rag efficiency: A comparative analysis of rag methods, Natural Language Processing 31 (2024) 1–25. doi:10.1017/nlp.2024.53.

[12] D. Rau, S. Wang, H. Déjean, S. Clinchant, J. Kamps, Context embeddings for efficient answer generation in retrieval-augmented generation, in: Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining, 2025, pp. 493–502.

[13] J.-S. Park, S.-M. Park, Llm-based question generation learning system for improve users' literacy skills, The Journal of the Korea institute of electronic communication sciences 19 (2024) 1243–1248.

[14] K. Li, Y. Zhang, Planning first, question second: An LLM-guided method for controllable question generation, in: L.-W. Ku, A. Martins, V. Srikumar (Eds.), Findings of the Association for Computational Linguistics: ACL 2024, Association for Computational Linguistics, Bangkok, Thailand, 2024, pp. 4715–4729. URL: https://aclanthology.org/2024.findings-acl.280/. doi:10.18653/v1/2024.findings-acl.280.

[15] O. Khattab, M. Zaharia, Colbert: Efficient and effective passage search via contextualized late interaction over bert, 2020. URL: https://arxiv.org/abs/2004.12832. arXiv:2004.12832.

[16] L. Wang, N. Yang, X. Huang, L. Yang, R. Majumder, F. Wei, Multilingual e5 text embeddings: A technical report, 2024. URL: https://arxiv.org/abs/2402.05672. arXiv:2402.05672.

[17] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, M. Zhou, Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, 2020. URL: https://arxiv.org/abs/2002.10957. arXiv:2002.10957.

[18] D. Zhang, J. Li, Z. Zeng, F. Wang, Jasper and stella: distillation of sota embedding models, 2025. URL: https://arxiv.org/abs/2412.19048. arXiv:2412.19048.