Polito Team at SemEval-2026 Task 8: Scaling Multi-Turn RAG: High-Performance Parallelized Pipeline for the MTRAG Benchmark

Murat Çelik* **Nejla Dinçer*** **Can Ersoy*** **Mert Toprak***
Barış Tan Ünal* **Riccardo Coppola†** **Flavio Giobergia†**
Politecnico di Torino
* {s338212,s343290,s336887,s336966,s338210}@studenti.polito.it
† {firstname.lastname}@polito.it

Abstract

Recently, Retrieval-Augmented Generation (RAG) has become a significant task in Large Language Models (LLMs). In multi-turn RAG, a good system must overcome the challenges of maintaining context as the dialogue turns progress and manage the issue of generating answers based on conversation history. In this work, we address the MTRAGEval task 8 at SemEval-2026, by presenting a high-performance, parallelised Multi-Turn RAG pipeline designed to address three subtasks: Retrieval (Subtask A), Generation (Subtask B), and End-to-End RAG (Subtask C). Our methodology utilises a Streamlit framework that allows users to embed diverse corpora with varying vector spaces and embedding models, facilitating configuration for each task based on its nature. Some key experiments focus on the performance of different vector databases and embedding models, the necessity of LLM-based query rewriting (QR) for non-standalone questions, the use of different rerankers, and the scale and performance of the selected LLM for answer generation. We conclude that a configuration utilising query rewriting along with reranking delivers the best results. The code is available on GitHub https://github.com/merttoprak1/MTRAGEval-Evaluating-Multi-Turn-RAG-Conversations.

1 Introduction

LLMs have revolutionized text generation tasks, but the increasing diversity of models and architectures makes adopting the relevant model for a specific task increasingly challenging (Pierri et al., 2025). Instead, a common approach is to use a general model and provide the relevant context to address a user query. However, these models have been shown to struggle with context preservation, hallucinations, and generating answers grounded in a corpus. Retrieval-Augmented Generation (RAG) mitigates these issues by grounding responses in relevant external passages. While single-turn RAG is well established, Multi-Turn RAG is more challenging, as it requires generating responses within an evolving conversation. The model must track dialogue history and adapt its retrieval strategy across turns.

Multi-turn conversations are particularly difficult due to the non-standalone nature of human dialogue. Users often rely on references such as "its address?" or "the previous one", making later questions ambiguous without prior context. Additionally, RAG systems must determine whether a question is answerable given the available corpus and decline when necessary to avoid hallucinations.

To address these challenges, the MTRAGEval task (Rosenthal et al., 2026b) at SemEval-2026 defines three subtasks:

Subtask A (Retrieval): Evaluate the ability to retrieve gold-standard passages for the final turn of a conversation.

Subtask B (Generation): Assess the ability to produce a faithful answer given perfect reference passages.

Subtask C (End-to-End RAG): Evaluate the full pipeline, where the system performs retrieval before generating the final response.

In this work, we explore a solution to Multi Turn RAG that considers different embedding models, vector databases, query rewriting strategies, reranking policies, and multiple LLMs for generation.

2 Related Work

Multi-Turn RAG Evaluation The rapid progress of LLMs has motivated the development of robust evaluation benchmarks. MT-Bench (Zheng et al., 2023) assesses multi-turn conversational ability using LLM-as-a-judge protocols, but does not explicitly target retrieval-augmented generation challenges. In contrast, the MTRAG Benchmark (Katsis et al., 2025) focuses on

Proceedings of the The 20th International Workshop on Semantic Evaluation (2026), pages 2447–2455

multi-turn RAG settings, requiring systems to retrieve relevant evidence and correctly handle unanswerable queries. Unlike traditional single-turn QA datasets where answers are guaranteed, MTRAG evaluates a model's ability to abstain when the retrieved context is insufficient, thus directly testing hallucination robustness.

Retrieval Strategies Retrieval is central to RAG systems. Sparse methods such as BM25 (Robertson and Zaragoza, 2009) rely on lexical matching, while Dense Retrieval (Karpukhin et al., 2020) maps queries and documents into a shared embedding space to capture semantic similarity. Many modern pipelines adopt a two-stage "Retrieve-and-Rerank" framework, where an initial retriever selects candidate passages that are subsequently re-scored by a Cross-Encoder (Nogueira and Cho, 2019) to improve precision.

Query Rewriting in Conversational AI In multi-turn settings, user queries often contain coreferences or elliptical expressions, making them unsuitable for direct retrieval. Query rewriting mitigates this issue by generating standalone queries conditioned on the conversation history (Vakulenko et al., 2021), improving alignment between conversational inputs and document retrieval mechanisms.

3 Methodology

Our system adopts a modular, component-based architecture designed for flexibility and high performance. Figure 1 summarises the main components of the pipeline, which consists of sequential stages:

User Input Processing: The system accepts a file containing a list of objects, each comprising a raw user *query* along with the *conversation history*,
Query Rewriting: An LLM-based rewriter transforms the context-dependent query into a standalone search query,
Embedding Generation: The rewritten query is converted into a dense vector embedding,
Vector Retrieval: A high-speed vector database (Qdrant or Faiss) retrieves the top-K most similar document chunks,
Reranking: A cross-encoder model re-scores the retrieved candidates to improve precision, filtering down to the top-N context passages,
Generation: The final context is fed into the LLM generator to produce the answer.

3.1 Subtask A: Retrieval Only

Subtask A isolates the retrieval subsystem. Given a user query and conversation history, the system must retrieve the specific passages that contain the answer. An important aspect in multi-turn conversations is detecting the intent of elliptical or coreferential queries (e.g., "Why did he do that?"). We employ a generative Query Rewriter that leverages the conversation history to resolve these ambiguities. The rewriter is prompted to produce a fully specified, standalone question that captures the user's original intent without requiring external context, which is essential for effective vector similarity search.

For the choice of vector search backends, we evaluated two alternatives, *FAISS* (Douze et al., 2024) and *Qdrant* (qdrant/qdrant Contributors, 2026). FAISS is a library for efficient similarity search and dense vector clustering, providing a lightweight, in-memory solution suitable for rapid prototyping. Qdrant, instead, is a production-grade vector database that supports more advanced features.

To balance recall and precision, we implemented a two-stage retrieval process. In the first stage, the vector database retrieves a broad set of candidates (e.g., Top-$K = 20$) based on cosine similarity. In the second stage, a Cross-Encoder model (such as *BGE-Reranker* (Chen et al., 2024) or *MS-MARCO* (Bajaj et al., 2018)) performs a computationally intensive re-scoring of these candidates. This reranking step is vital for filtering out "hard negatives"—documents that are semantically similar but factually irrelevant—thereby presenting the generator with only the highest-quality context. We evaluate both Top-$N = 5$ and Top-$N = 10$ after reranking to measure the trade-off between retaining enough evidence and avoiding noisy context.

3.2 Task B: Generation with Reference Passages

For Task B generation, our main goal is to generate relevant answers to the last question in the given set of "tasks". A *task* is defined as a conversation turn containing all previous turns together with the last user question (e.g., the task created for turn i includes all user and agent questions/responses for the first $i - 1$ turns plus the user question for turn i) (Katsis et al., 2025).

The full prompt used for the generation tasks contains 4 components:

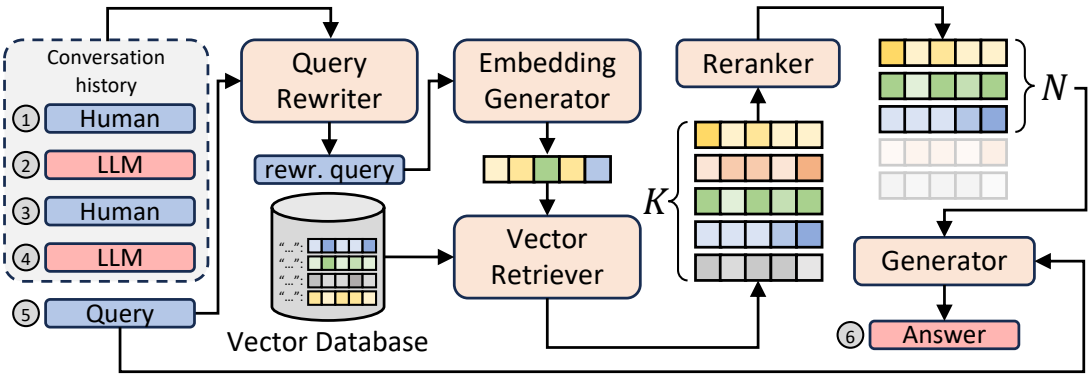

Figure 1: High-level architecture of the proposed solution

1. **System Prompt:** defines the assistant's role and instructions (see Appendix B);

2. **Conversation History:** all previously exchanged messages;

3. **Current Question:** the latest user query;

4. **Context:** reference passages for generation (as extracted in the retrieval phase, ordered in descending order of relevance, and numbered).

The prompt is used as input to an LLM to generate an answer for the user's question. We tested a variety of LLMs. We selected models from different companies and different sizes to assess the variance in generated results. More specifically, we considered `gpt-oss-20b` from OpenAI, `gemma-3-12b` from Google, `deepseek-r1-0528-qwen3-8b` from DeepSeek, `ministral-3-14b-reasoning` and `ministral-3-3b` from MistralAI (all names are as reported on HuggingFace).

3.3 Subtask C: Generation with Retrieved Passages

Subtask C combines retrieval and generation in a single pipeline. Given a conversation and a document corpus, the system first retrieves relevant passages and then generates an answer based only on those passages.

We combine the solutions from Subtasks A and B using QR and Qdrant as vector databases, with $K = 20$ for the initial passage retrieval and $N = 5, 10$ for the identification of the best results after reranking. The full prompt and adopted generative models are defined according to 3.2.

4 Experimental Results

All experiments were conducted on a heterogeneous compute cluster, using Apple M3 Pro silicon for local inference and NVIDIA T4 GPUs for more intensive reranking tasks. We evaluated the retrieval pipeline (Subtask A) using Recall@k (R@k) and nDCG@k to measure context relevance and ranking quality. For generation quality (Subtasks B & C), we computed three critical metrics as given in the task (Katsis et al., 2025): $\mathbf{RL_F}$ **(Faithfulness)**, a binary score indicating whether the answer is derived *solely* from the provided context to prevent hallucinations, $\mathbf{RB_{alg}}$, the harmonic mean of three algorithmic metrics: Bert-Recall (Bert-Rec), Bert-K-Precision (Bert-K-Prec), and Rouge-L, and $\mathbf{RB_{llm}}$, a Reference-Based LLM judge adapted from RAD-Bench. All of these metrics are then conditioned on **IDK LLM judge** that first determines if the response contains an answer.

To isolate the contribution of each component, we defined a hyperparameter search space across three dimensions. For **Embeddings**, we compared `all-minilm:l6-v2`, `nomic-embed-text`, and `jina-embeddings-v2`. For **Reranking**, we evaluated `BGE-v2-M3`, `Mxbai-Rerank-xsmall`, and `cross-encoder/ms-marco`. Finally, we tested varying **Retrieval Depths**, specifically identifying the Top-K ($K = 20, 30$) candidates from the vector store and reranking the Top-N ($N = 5, 10$) for final context generation.

Finally, we note that the dataset provided for the task spans four domains: general knowledge (ClapNQ), cloud documentation (Cloud), financial discussions (FiQA), and government portals (Govt), covering diverse linguistic styles and domain-specific terminology.

4.1 Subtask A: Retrieval Performance

We evaluated the retrieval pipeline based on nDCG@k and Recall@k at varying depths ($k = 3, 5$). Our primary objective was to maximize the relevance of the context provided to the generation phase while maintaining low latency. We note that Qdrant significantly outperformed FAISS under the same standard default configurations (nDCG@3: 0.227 vs 0.050). We hypothesize that this performance disparity likely originates from differences in default index construction: Qdrant utilizes highly optimized default parameters for its Hierarchical Navigable Small World (HNSW) graph and natively aligns with cosine similarity. In contrast, FAISS is a lower-level library that requires explicit manual tuning of its index structures and distance metrics to achieve optimal recall. Consequently, we adopted Qdrant for all remaining experiments to ensure that the retrieval backend was not a bottleneck. We established a **Baseline** configuration using Qdrant, `all-minilm-l6-v2` embeddings, and no re-ranking, no QR, which yielded an nDCG@3 of 0.249.

Query Rewriting Strategies Given the conversational nature of MTRAG, where queries often depend on previous turns, direct retrieval often fails. We compared two models for Query Rewriting: `gemma-3-12b` and `gpt-oss-20b`. While enabling QR with the smaller `gemma-3-12b` model improved the baseline nDCG@3 to 0.268 (+7.6%), the larger `gpt-oss-20b` model achieved 0.293 (+17.6%). This substantial gap suggests that the reasoning capabilities required to disambiguate complex multi-turn references do scale with model size. In particular, the rewriter must recover omitted entities, resolve pronouns, and preserve constraints introduced in earlier turns; small errors in this step change the semantic target of the vector search and therefore cannot be corrected reliably by later reranking. Consequently, we utilised `gpt-oss-20b` for all subsequent retrieval experiments to isolate downstream architectural effects. Even though we did not observe any loss of performance in terms of score when we utilize QR, there is a slight increase in time required to complete the retrievals since with QR another step is included.

Embedding Models We evaluated the semantic capture capabilities of four embedding models. As shown in Table 1, larger models do not strictly guarantee better performance on the MTRAG dataset. The state-of-the-art `jina-embeddings-v2` achieved the highest accuracy (0.332), but the compact `all-minilm` model remained extremely competitive (0.329). Given the negligible performance difference ($< 1\%$), `all-minilm` represents a superior efficiency-performance trade-off for latency-sensitive applications. This result is plausible because MTRAG retrieval often depends on matching short, rewritten questions to compact answer-bearing passages, where robust sentence-level similarity can be more important than broad multilingual or long-context capacity.

Embedding Model	nDCG@3	R@3
Snowflake-Arctic (137m)	0.059	0.058
Nomic-Embed-Text	0.227	0.219
All-MiniLM-L6-v2	0.329	0.311
Jina-Embeddings-v2	**0.332**	**0.313**

Table 1: Impact of Embedding Models on retrieval performance. Experiments utilized Qdrant and BGE-v2-M3 (Top-10) reranking.

Conversely, `snowflake-arctic` suffered a catastrophic drop (0.059). We suspect this sharp decrease in performance is due to a mismatch in query expectations. While this model is highly optimized for matching short, direct questions to longer documents, our query rewriter (`gpt-oss-20b`) often generates long and highly detailed queries. The rewriter does this to clarify conversational ambiguities, such as replacing words like "it" or "the previous one" with the actual specific subjects mentioned in earlier turns. This increase in query length, combined with the model's difficulty in processing the highly specialized jargon found in the Cloud and FiQA domains, likely pushes the text outside the model's optimal working range, causing the semantic matching to fail.

Impact of Reranking Architectures The second-stage reranking step aims to mitigate the precision limitations of dense vector retrieval. We experimented with cross-encoder architectures to re-score the top-K documents retrieved by the vector store.

As shown in Table 2, modern rerankers significantly outperform older architectures. While `MS-Marco` provided negligible gains over the non-reranked pipeline, `BGE-v2-M3` achieved the highest performance across all metrics. We further investigated the final reranked depth—the number of passages retained after cross-encoder scoring. Counter-intuitively, keeping only the top-5 pas-

Reranker Model	QR	Depth	nDCG@3	R@3
None (Baseline)	No	-	0.249	0.236
None	Yes	-	0.293	0.275
MS-Marco-MiniLM	Yes	10	0.293	0.275
Mxbai-xsmall-v1	Yes	10	0.317	0.300
BGE-v2-M3	Yes	10	0.329	0.311
BGE-v2-M3	**Yes**	**5**	**0.342**	**0.326**

Table 2: Comparison of Reranking models and depths. All experiments used Qdrant, all-minilm-l6-v2 embeddings, and `gpt-oss-20b` for Query Rewriting.

LLM Model	H.Mean	$\mathbf{RL_F}$	$\mathbf{RB_{llm}}$	$\mathbf{RB_{alg}}$
`gemma-3-12b`	**0.59**	**0.82**	0.69	**0.42**
`gpt-oss-20b`	0.55	0.74	**0.74**	0.36
`ministral-3-14b-R`	0.54	0.76	0.68	0.36
`DS-R1-Q3-8B`	0.52	0.75	0.66	0.35
`ministral-3-3b`	0.51	0.79	0.71	0.31

Table 3: Subtask B generation performance including Faithfulness and Reference-Based metrics. `ministral-3-14b-R` refers to `ministral-3-14b-reasoning`, `DS-R1-Q3-8B` refers to `deepseek-r1-0528-qwen3-8b`.

sages yielded higher early-ranking performance (nDCG@3: 0.342, nDCG@5: 0.384) than keeping the top-10 (nDCG@3: 0.329, nDCG@5: 0.365; see Table A.1), even though the top-10 setting can recover more total relevant passages at larger cutoffs. This indicates a precision-recall trade-off: MTRAG questions usually target a small amount of specific evidence from the latest conversational turn, so the most important behaviour is to place the answer-bearing passage in the first few ranks rather than to maximise the size of the retained context. Lower-ranked candidates often share entities or topic words with the query but answer a different aspect of the conversation. These near-topic negatives are difficult for cross-encoders because lexical and entity overlap can look relevant even when the passage is not answer-bearing. As a result, expanding from five to ten passages introduces additional distractors that can reduce early nDCG and, in the generation setting, can also make the answer model condition on irrelevant evidence. The Top-5 setting therefore acts as a noise-control mechanism: it slightly sacrifices breadth, but improves the precision of the context actually exposed to the downstream pipeline.

In the challenge, our system had a Retrieval Score (nDCG@5) of 0.4048, using the test dataset proposed in the MTRAG-UN (Rosenthal et al., 2026a), ranking 23 out of 38. A detailed comparison can be seen in C.2.

4.2 Subtask B: Generation Performance

We present the results of our Subtask B evaluation in Table 3. The general consideration is, as expected, that the performance increases with model size. There are, however, situations where it is not the case.

`gemma-3-12b` is the best performing model among the five models evaluated based on the harmonic mean of $\mathbf{RL_F}$, $\mathbf{RB_{llm}}$ and $\mathbf{RB_{alg}}$ scores, followed by `gpt-oss-20b`. This shows that generation quality in grounded RAG is not determined only by parameter count. The task rewards models that can follow the provided passages closely, avoid unsupported elaboration, and produce concise answers matching the reference style. A larger model may produce more fluent or expansive responses, but this can hurt the harmonic mean when the extra content is not explicitly supported by the reference passages.

When considering faithfulness to the provided context, `gemma-3-12b` ranks first ($\mathbf{RL_F}$ of 0.82). To effectively control hallucinations, our structured prompting strategy strictly constrains generation to the provided reference passages. However, our results indicate that structural differences in model instruction tuning yield varying degrees of context adherence. Specifically, while reasoning oriented architectures such as `deepseek-r1-0528-qwen3-8b` demonstrate advanced logical deduction, they frequently suffer from over reasoning, appending inferential steps that lack direct textual support. Conversely, larger generalist models such as `gpt-oss-20b` exhibit "knowledge leakage", bypassing the prompt's constraints to prioritize their memory over retrieved evidence when passages are incomplete. Interestingly, the compact `ministral-3-3b` shows that high faithfulness can be achieved even for small models ($\mathbf{RL_F}$ of 0.79), suggesting that instruction tuning and context adherence are as important as scale for this benchmark. Balancing these generation dynamics, our final submitted system achieved a generation score (harmonic mean of $\mathbf{RL_F}$, $\mathbf{RB_{llm}}$ and $\mathbf{RB_{alg}}$) of 0.6377, using the test dataset proposed in the MTRAG-UN (Rosenthal et al., 2026a), ranking 16 out of 26. A detailed comparison can be seen in C.2.

LLM Model	Embedding Model	Reranker	TopK	H.Mean	$\mathbf{RL_F}$	$\mathbf{RB_{llm}}$	$\mathbf{RB_{alg}}$
gemma-3-12b	Jina-Embed-v2	BGE-Reranker-v2	10	**0.53**	0.84	0.63	0.34
gemma-3-12b	All-MiniLM-L6	BGE-Reranker-v2	5	0.52	**0.85**	0.59	**0.35**
gpt-oss-20b	All-MiniLM-L6	BGE-Reranker-v2	10	0.50	0.81	**0.67**	0.30
gpt-oss-20b	Jina-Embed-v2	BGE-Reranker-v2	10	0.49	0.77	0.65	0.30
gpt-oss-20b	All-MiniLM-L6	Mxbai-XS	10	0.48	0.79	0.65	0.29
gpt-oss-20b	All-MiniLM-L6	BGE-Reranker-v2	5	0.48	0.76	0.61	0.30
gpt-oss-20b	Nomic-Embed	BGE-Reranker-v2	10	0.39	0.62	0.50	0.24

Table 4: Subtask C performance comparison according to different settings, ranking based on Harmonic Mean, calculated using $\mathbf{RL_F}$, $\mathbf{RB_{llm}}$, $\mathbf{RB_{alg}}$ metrics.

4.3 Subtask C: End-to-End Performance

In Subtask C the generator must rely solely on the noisy context provided by the retrieval system. Table 4 presents the performance of our top configurations. While running settings for this part, we experimented with the best results of Subtask A and Subtask B, since Subtask C is a combination of them.

Similarly to Subtask B, we obtained the best ranking with the `gemma-3-12b` model, which outperformed the larger, `gpt-oss-20b`.

Performance in Subtask C varies notably with the choice of embeddings and the reranker. **BGE-Reranker-v2** consistently achieved the highest scores, while **All-MiniLM-L6** provided strong overall embedding performance; however, the best pipeline used **Jina-Embed-v2**. This suggests that the retrieval stack should be tuned jointly rather than component by component: the embedding model controls the candidate pool, while the reranker controls which of those candidates survive into the prompt. A slightly stronger embedding model can therefore improve end-to-end generation even when its retrieval-only advantage is small, because it gives the reranker better candidates to order.

Retrieval errors directly propagate to generation. We observe generally **low Algorithmic Overlap ($\mathbf{RB_{alg}}$)** (0.24-0.35), reflecting lexical divergence from the reference when the retriever misses the gold passages. In contrast, **Faithfulness ($\mathbf{RL_F}$)** remains relatively high (up to 0.85), indicating that models adhere to the retrieved context even when it is incomplete or incorrect. We hypothesize this behavior is a direct result of RAG-specific instruction tuning: these models are aggressively trained to suppress their internal parametric memory in favor of the provided text. Consequently, they become overly obedient to the retrieved context. This dynamic confirms that retrieval recall, rather than generation capability, acts as the absolute bottleneck in the end-to-end setting. Any upstream retrieval failure guarantees a faithful but incorrect generation, imposing a strict ceiling on the overall pipeline's performance.

In the challenge, our system had a generation score (harmonic mean of $\mathbf{RL_F}$, $\mathbf{RB_{llm}}$ and $\mathbf{RB_{alg}}$) of 0.4848, using the test dataset proposed in the MTRAG-UN (Rosenthal et al., 2026a), ranking 21 out of 29. A detailed comparison can be seen in C.2.

5 Conclusion

We presented a high-performance, modular RAG pipeline designed for the MTRAG benchmark. Our system evaluates three distinct subtasks: Retrieval (Subtask A), Generation (Subtask B), and End-to-End RAG (Subtask C). Our work demonstrates that Multi-Turn RAG requires robust LLM-based QR to resolve ambiguities and an optimised two-stage "Retrieve-and-Rerank" architecture to maximise precision. Interestingly, our experiments reveal that smaller, state-of-the-art models such as `gemma-3-12b` can outperform larger baselines, such as `gpt-oss-20b`.

We believe that future research should focus on handling "unanswerable" questions to reduce hallucination rates. Additionally, we plan to explore synthetic data augmentation using the MTRAG-S framework to improve the system's robustness across diverse domains.

References

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. Ms marco: A human generated machine reading comprehension dataset. *Preprint*, arXiv:1611.09268.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Yannis Katsis, Sara Rosenthal, Kshitij Fadnis, Chulaka Gunasekara, Young-Suk Lee, Lucian Popa, Vraj Shah, Huaiyu Zhu, Danish Contractor, and Marina Danilevsky. 2025. Mtrag: A multi-turn conversational benchmark for evaluating retrieval-augmented generation systems.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Francesco Pierri, Anna Bernasconi, Flavio Giobergia, Claudio Savelli, Elena Baralis, Luca Cagliero, and Stefano Ceri. 2025. A conceptual map for exploring the landscape of large language models. *IEEE Access*.

qdrant/qdrant Contributors. 2026. Qdrant: High-performance, massive-scale vector database and vector search engine. `https://github.com/qdrant/qdrant`. Version 1.17.0, accessed February 2026.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3:333–389.

Sara Rosenthal, Yannis Katsis, Vraj Shah, Lihong He, Lucian Popa, and Marina Danilevsky. 2026a. Mtrag-un: A benchmark for open challenges in multi-turn rag conversations. *Preprint*, arXiv:2602.23184.

Sara Rosenthal, Vraj Shah, Yannis Katsis, and Marina Danilevsky. 2026b. Semeval-2026 task 8: Mtrageval: Evaluating multi-turn rag conversations. In *Proceedings of the 20th International Workshop on Semantic Evaluation (SemEval-2026)*, San Diego, California. Association for Computational Linguistics.

Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anotha. 2021. Question rewriting for conversational question answering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 355–363.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Sanskar Aketi, Mayur J. Rice, Bei Chen, Dacheng Tao, Gunho Park, Ion Stoica, Michael I. Jordan, and Xuanzhe Liu. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

Appendix

A Detailed Retrieval and Generation Experiments

ID	QR	VectorDB	QR Model	Embedding	Reranker	Depth	nDCG@1	nDCG@3	nDCG@5	nDCG@10	R@1	R@3	R@5	R@10
1	YES	Qdrant	gpt-oss-20b	All-MiniLM-L6	BGE-v2-M3	5	0.360	0.342	0.384	0.382	0.149	0.326	0.431	0.431
2	YES	Qdrant	gpt-oss-20b	Jina-Embed-v2	BGE-v2-M3	10	0.349	0.332	0.364	0.419	0.148	0.313	0.397	0.529
3	YES	Qdrant	gpt-oss-20b	All-MiniLM-L6	BGE-v2-M3	10	0.348	0.329	0.365	0.418	0.143	0.311	0.406	0.533
4	YES	Qdrant	gpt-oss-20b	All-MiniLM-L6	BGE-v2-M3	10	0.335	0.320	0.353	0.406	0.139	0.303	0.392	0.518
5	YES	Qdrant	gpt-oss-20b	All-MiniLM-L6	Mxbai-XS	10	0.337	0.317	0.351	0.403	0.139	0.300	0.389	0.511
6	YES	Qdrant	gemma-3-12b	All-MiniLM-L6	BGE-v2-M3	5	0.326	0.317	0.359	0.357	0.133	0.302	0.405	0.405
7	YES	Qdrant	gpt-oss-20b	All-MiniLM-L6	Jina-Reranker-v2	10	0.334	0.316	0.350	0.407	0.140	0.300	0.388	0.522
8	YES	Qdrant	gpt-oss-20b	All-MiniLM-L6	MS-Marco	10	0.320	0.293	0.324	0.369	0.134	0.275	0.356	0.464
9	YES	Qdrant	gpt-oss-20b	All-MiniLM-L6	None	-	0.320	0.293	0.324	0.369	0.134	0.275	0.356	0.463
10	YES	Qdrant	gemma-3-12b	All-MiniLM-L6	BGE-v2-M3	10	0.303	0.290	0.328	0.381	0.124	0.274	0.368	0.495
11	YES	Qdrant	gemma-3-12b	All-MiniLM-L6	Mxbai-XS	10	0.308	0.286	0.326	0.376	0.126	0.268	0.366	0.483
12	NO	Qdrant	-	All-MiniLM-L6	BGE-v2-M3	10	0.285	0.269	0.300	0.346	0.120	0.255	0.334	0.443
13	YES	Qdrant	gemma-3-12b	All-MiniLM-L6	None	-	0.284	0.268	0.295	0.344	0.117	0.253	0.324	0.438
14	NO	Qdrant	-	All-MiniLM-L6	None	-	0.262	0.249	0.269	0.308	0.113	0.237	0.293	0.385
15	YES	Qdrant	gpt-oss-20b	Nomic-Embed	BGE-v2-M3	10	0.234	0.227	0.256	0.301	0.099	0.219	0.292	0.398
16	YES	Qdrant	gemma-3-12b	Nomic-Embed	BGE-v2-M3	10	0.232	0.230	0.253	0.297	0.095	0.224	0.285	0.389
17	NO	Qdrant	-	Nomic-Embed	BGE-v2-M3	10	0.227	0.210	0.233	0.273	0.100	0.198	0.256	0.350
18	YES	Qdrant	gemma-3-12b	Nomic-Embed	None	-	0.224	0.207	0.230	0.264	0.092	0.197	0.257	0.339
19	YES	Qdrant	gpt-oss-20b	Nomic-Embed	None	-	0.216	0.205	0.230	0.267	0.091	0.197	0.260	0.350
20	NO	Qdrant	-	Nomic-Embed	None	-	0.211	0.188	0.208	0.240	0.094	0.174	0.227	0.300
21	YES	Qdrant	gpt-oss-20b	Snowflake-137m	BGE-v2-M3	10	0.058	0.059	0.067	0.083	0.024	0.058	0.078	0.114
22	YES	Faiss	gpt-oss-20b	Nomic-Embed	BGE-v2-M3	10	0.039	0.050	0.078	0.194	0.014	0.050	0.106	0.397

Table A.1: Full Experimental Results for Subtask A (Retrieval), sorted by nDCG@3 performance. Abbreviations: QR (Query Rewriting).

B Default System Prompt Used for Tasks B and C

System Prompt Used for Subtasks B and C

```
You are an expert generative AI assistant in a RAG pipeline.

When given a user query:
1. Always prioritize *only information from the retrieved documents* included
in the context.
2. Do not hallucinate facts or invent details not supported by those documents.
3. If the retrieved context lacks necessary information, acknowledge it clearly.
4. If there is no answer in the retrieved contents answer like this: I'm sorry, but
I don't have the answer to your question.
5. Follow these steps:
   a. Briefly summarize the relevant retrieved content linked to the query.
   b. Integrate that content into a coherent answer.
   c. Cite or reference the source document or snippet when appropriate.
6. Use precise, clear language and format answers for readability.
7. Do not write for the indicating like, answer, source and any others
Answer the user's question now using the retrieved context.
```

C Comparison of Results with Baselines and Leaderboard

Subtask	System	Type	Main Metric	Score	Rank
A	AILS-NTUA	Leaderboard best	nDCG@5	0.578	1/38
A	Elser	Official baseline	nDCG@5	0.450	-
A	**Polito Team**	**Our system**	**nDCG@5**	**0.405**	**23/38**
A	BGE-base 1.5	Official baseline	nDCG@5	0.270	-
B	RaguTeam	Leaderboard best	H-Mean	0.783	1/26
B	GPT-OSS-120B	Official baseline	H-Mean	0.639	-
B	**Polito Team**	**Our system**	**H-Mean**	**0.638**	**16/26**
B	GPT-OSS-20B	Official baseline	H-Mean	0.590	-
C	GenAIus	Leaderboard best	H-Mean	0.586	1/29
C	Qwen-30B-Think	Official baseline	H-Mean	0.537	-
C	GPT-OSS-20B	Official baseline	H-Mean	0.514	-
C	**Polito Team**	**Our system**	**H-Mean**	**0.485**	**21/29**

Table C.2: Comparison of Results with Leaderboard and Baselines across subtasks A, B, and C