Experimental Study of Retrieval Augmented Generation for Multi-Hop Question Answering

Anton Risch

24th October 2025

Abstract

This report presents an experimental study evaluating Retrieval Augmented Generation (RAG) systems for Multi-Hop Question Answering (QA). The multi-hop task is a significant challenge in Open-Domain Question Answering (ODQA), requiring systems to retrieve and synthesise facts from multiple unknown documents within a large corpus. This analysis was conducted in two stages: first, an evaluation of four dense retrieval models (embedding-based) on a 2,556-query multi-hop corpus; second, an end-to-end RAG pipeline evaluation combining the top two retrievers with four instruction-tuned Large Language Models (LLMs) (7-8B parameters) representing diverse architectures. Results from Stage 1 showed that Ranker B (BAAI/bge-base-en-v1.5) achieved 76.88% retrieval effectiveness (Hits@10), a 10% improvement over the next-best model. In Stage 2, LLM choice was the dominant factor in system performance, varying F1 scores by 80% (Llama-2: F1=0.23 vs Qwen2-7B: F1=0.42), while the 10% difference in retriever quality contributed less than 1% to final F1 scores. This demonstrates that RAG effectiveness for the QA task depends more on reader-stage model configuration than on retriever-stage quality, particularly once retrieval recall exceeds a ∼70% threshold. Ultimately, Llama-2 exhibited a 99.3% hallucination rate on unanswerable queries, whereas Qwen2 correctly abstained 75% of the time.

1 Introduction

Multi-Hop Question Answering (QA) is a challenging task within Open-Domain Question Answering (ODQA). Unlike single-document QA, where relevant text is pre-selected, multi-hop QA requires systems to retrieve necessary information from large document corpora where the location of relevant passages is unknown, then synthesise facts across multiple documents to construct coherent answers.

Large Language Models (LLMs) answer questions using knowledge encoded in their internal parameters that are learned from predetermined training corpora. However, this approach presents critical limitations: knowledge cutoff dates restrict access to up-to-date information, and the absence of external grounding can lead to hallucinations, especially for recent events or specialised domains.

Retrieval Augmented Generation (RAG) addresses these limitations by implementing a two-stage retriever-reader pipeline. First, a retriever searches the document collection D and selects the k most relevant passages ($P_1 \ldots P_k$) for a given query Q. Then, a reader (usually an instruction-tuned LLM) generates an answer A conditioned on the retrieved context. This architecture grounds generation in external, citable documents rather than relying purely on parametric knowledge.

However, RAG system design involves non-trivial component selection. The combination of the highest-performing retriever and highest-performing LLM does not necessarily yield the best end-to-end system in practice, as component interactions can be non-linear and unpredictable.

I created some research questions to guide the study and its analysis:

1. Which dense retrieval models are most effective for multi-hop QA?
2. How do different instruction-tuned LLM architectures perform as readers in RAG systems?
3. Do retriever and LLM choices act independently, or do certain retriever-LLM pairings work better (or worse) together?

To answer these questions, this study employed a two-stage experimental design. Stage 1 evaluated four embedding-based dense retrievers on a 2,556-query multi-hop corpus. Stage 2 combined the top two retrievers with four instruction-tuned LLMs (Llama-2-7b, Llama-3-8B, Mistral-7B, Qwen2-7B), representing diverse architectures, to yield eight complete RAG pipelines (2 retrievers $\times$ 4 LLMs = 8 configurations).

When selecting models, I prioritised architectural diversity but also considered computational constraints (44-hour GPU quota, as well as an additional 16GB VRAM GPU I supplied myself). This approach allowed for a broad evaluation of LLMs across different retriever backbones, highlighting how these components interact, rather than focusing solely on finding the absolute best retriever.

2 Background and Related Work

2.1 Dense Retrieval Methods

Dense retrieval systems encode queries and documents into continuous vector representations (embeddings), with their similarity measured via cosine distance (Jurafsky & Martin, 2024). Unlike lexical methods (e.g., BM25) that rely on keyword matching, dense retrieval captures semantic relationships. For example, "car" and "automobile" would receive similar embeddings despite having no overlapping words – unlike BM25 which would not match them due to lack of shared tokens. Modern approaches employ sentence transformers with bi-encoder architecture: queries and documents are encoded independently, enabling pre-computation and efficient nearest-neighbor search over large corpora.

2.2 Large Language Models for Question Answering

Instruction-tuned LLMs evolve from base language models through supervised fine-tuning (SFT) and reinforcement learning with human feedback (RLHF) (Jurafsky & Martin, 2024). Unlike base models trained for next-token prediction, instruction-tuned variants are optimised for dialogue and task completion: they understand instructions, format responses appropriately, and can abstain from answering when appropriate.

Modern 7-8B parameter models demonstrate strong QA performance while remaining deployable on consumer GPUs. Architectural innovations include Grouped-Query Attention (GQA) for improved inference efficiency (Meta AI, 2024), extended vocabulary for better tokenisation (Mistral AI, 2024), and exceptionally long context windows (Qwen Team, 2024). Different model families reflect varied training philosophies: Llama emphasises RLHF alignment, Mistral focuses on efficient

architecture, and Qwen targets multilingual capability.

Prompt engineering is also critical: instruction format, context presentation, and output constraints significantly affect generation quality (Jurafsky & Martin, 2024). For factual QA, low temperature (0.1–0.2) encourages deterministic responses over potentially inaccurate creative outputs.

2.3 RAG System Architecture

RAG systems implement a two-stage pipeline: a retriever selects k candidate documents from corpus D based on query Q, then a reader (LLM) generates answer A conditioned on the retrieved context (Jurafsky & Martin, 2024). Formally: the retriever computes $P = \text{top-}k(D, Q)$, and the reader generates $A \sim P(A \mid Q, P)$.

This architecture addresses LLM limitations by grounding generation in external, updateable knowledge. However, performance is not simply the product of component quality. Component interactions are non-linear: retriever failure propagates (no relevant documents → impossible task for reader), but retriever success does not guarantee reader success (relevant documents present but LLM hallucinates or extracts incorrectly).

This creates an experimental challenge: evaluating components in isolation (Stage 1 retrieval metrics) may not predict end-to-end system performance (Stage 2 answer accuracy), creating the need for full pipeline evaluation across multiple component combinations.

3 Methodology

3.1 Dataset

The evaluation corpus contained 609 documents from a variety of news sources. They primarily consist of articles from sporting News (101), TechCrunch (97), The Verge (45), Polygon (44), and Fortune (25). Domain coverage was heavily skewed toward Technology and Sports, with smaller representation from Business, Entertainment, and Science/Health.

The query set comprised 2,556 multi-hop questions requiring 2–3 evidence documents per answer. Question types included: Comparison queries (856, 33.5%), Inference queries (816, 31.9%), Temporal queries (583, 22.8%), and Null queries (301, 11.8%) with insufficient information in the corpus. A gold standard was provided, including the correct answer, relevant document IDs, and question type label for each query.

3.2 Stage 1: First-Stage Retrieval

Tested models. Four dense retrieval models were evaluated, all pre-cached on the GPU server to conserve compute quota by eliminating download overhead. The models represent diverse design philosophies:

Ranker A (BAAI/llm-embedder): Optimised for LLM-compatible representations and integration with generative models (Zhang et al., 2023).

Ranker B (BAAI/bge-base-en-v1.5): General-purpose dense retrieval model with BERT-based bi-encoder architecture (Xiao et al., 2023).

Ranker C (sentence-transformers/all-MiniLM-L6-v2): Lightweight 6-layer distilled model optimised for inference speed (Reimers & Gurevych, 2019).

Ranker D (thenlper/gte-base): Trained on diverse data for general text embedding and cross-domain retrieval (Li et al., 2023).

Configuration and metrics. Retrieval configuration: top-$k = 10$ documents per query, cosine similarity metric, no query expansion or preprocessing beyond tokenization.

Evaluation metrics (Jurafsky & Martin, 2024): Hits@k (proportion of queries with $\geq$1 relevant documents in top-k), MAP@10 (Mean Average Precision at 10), MRR@10 (Mean Reciprocal Rank at 10).

3.3 Stage 2: RAG with LLMs

Selected retrievers. Based on Stage 1 results, the top two performing retrievers were selected for full RAG evaluation: Ranker B (BAAI/bge-base-en-v1.5, Hits@10 = 0.6647) and Ranker A (BAAI/llm-embedder, Hits@10 = 0.5348).

While reranking could potentially improve lower-performing retrievers (Ranker C: 0.5100, Ranker D: 0.4545), implementing reranking would require additional GPU time (1.5 hours vs 1 hour per run). Given the 44-hour GPU quota, I prioritised completing the full experimental pipeline (2 retrievers × 4 LLMs = 8 RAG evaluations) over exploring alternative Stage 1 architectures.

LLM models tested. Four instruction-tuned LLMs (7–8B parameters) representing diverse ar-

chitectures were evaluated, all pre-cached on the GPU server:

Llama-2-7b-chat-hf (7B): Baseline model with RLHF training, 4k context length, 2T training tokens (Touvron et al., 2023).

Meta-Llama-3-8B-Instruct (8B): Represents generational improvement with 15T+ training tokens, 8k context, and Grouped-Query Attention (GQA) for improved inference (Meta AI, 2024).

Mistral-7B-Instruct-v0.3 (7B): Alternative architecture with extended vocabulary (32k tokens) and function calling capabilities (Mistral AI, 2024).

Qwen2-7B-Instruct (7B): Features SwiGLU activation, attention QKV bias, and exceptionally long context support (131k tokens). Provides geographic and training diversity (Qwen Team, 2024).

RAG configuration. Prompt template: a structured prompt instructed the LLM to answer based solely on context, provide concise answers, and explicitly state "Insufficient Information" if the context was inadequate. Retrieved documents were concatenated with `----------` delimiters.

Generation params: temperature = 0.1, max tokens = 512, sampling enabled, padding token = `<eos>`.

Response parsing: each LLM required model-specific parsing due to different special tokens. Llama-2/Mistral split on `[/INST]`; Llama-3 on its structured assistant header markers; Qwen2 on `assistant:`. Parsing strategies were identified through iterative testing with `STAGING=True` on a random subset of queries, examining raw model outputs to determine correct delimiter tokens for each model family.

Evaluation metrics: Exact Match (EM), Precision (correct / (correct + incorrect)), Recall (correct / (correct + missed)), F1 Score (harmonic mean of Precision and Recall), and "Insufficient Info" Accuracy (correctly identifying unanswerable queries).

4 Experimental Setup

4.1 Infrastructure

The evaluation was conducted on an NVIDIA A10 GPU via university compute cluster with a 44-hour quota. The implementation utilised LlamaIndex 0.9 with Python 3.12, key libraries including transformers, pytorch, and sentence-transformers.

4.2 Execution Pipeline

The evaluation used a two-phase testing strategy (`STAGING=True` for sample validation, `STAGING=False` for full evaluation) to validate implementation correctness and conserve GPU quota.

Stage 1 (Retrieval):

1. Test all 4 retrievers locally with `STAGING=True` (sample data validation)
2. Run all 4 retrievers on GPU with `STAGING=False` (full 2,556 queries)
3. Evaluate using `evaluate_stage_1.py` to compute Hits@k, MAP@10, MRR@10
4. Select top 2 based on Hits@10 performance.

Stage 2 (RAG):

1. For each selected retriever, test each LLM with `STAGING=True` (verify response parsing), then run full evaluation with `STAGING=False`
2. Run 2 retrievers $\times$ 4 LLMs = 8 complete RAG pipelines
3. Evaluate using `evaluate_rag.py` to compute Precision, Recall, F1, Exact Match, and Insufficient Info Accuracy.

5 Results

5.1 Stage 1: Retrieval Performance

Overall metrics. Ranker B (bge-base) significantly outperformed all others across all metrics (76.88% Hits@10, 71.28% Hits@4), representing a 10% improvement over the second-best model (Ranker A, 69.99% Hits@10). The 18.1% gap in performance between the best and worst models (Ranker D: 64.95%) justified selecting Ranker B and Ranker A for Stage 2 evaluation. MAP@10 scores (0.23–0.29) indicate ranking quality limitations – relevant documents were retrieved but not consistently ranked highest. MRR@10 scores (0.50–0.66) show the first relevant document usually appeared around positions 2–3.

Variance analysis. The bar chart displays Hits@10 performance across all four rankers. Ranker B demonstrated consistent superiority, while the speed-optimised Ranker C (MiniLM-L6, 6 layers only) lost 10% accuracy to achieve faster inference. Ranker D (gte-base), despite targeting cross-domain generalisation, performed worst on the

Table 1: First-Stage Retrieval Results

Model	Hits@4	Hits@10	MAP@10	MRR@10
Ranker A (llm-embedder)	0.6154	0.6999	0.2459	0.5644
Ranker B (bge-base-en-v1.5)	0.7128	0.7688	0.2864	0.6561
Ranker C (all-MiniLM-L6-v2)	0.5700	0.6690	0.2364	0.5141
Ranker D (gte-base)	0.5567	0.6495	0.2290	0.5044

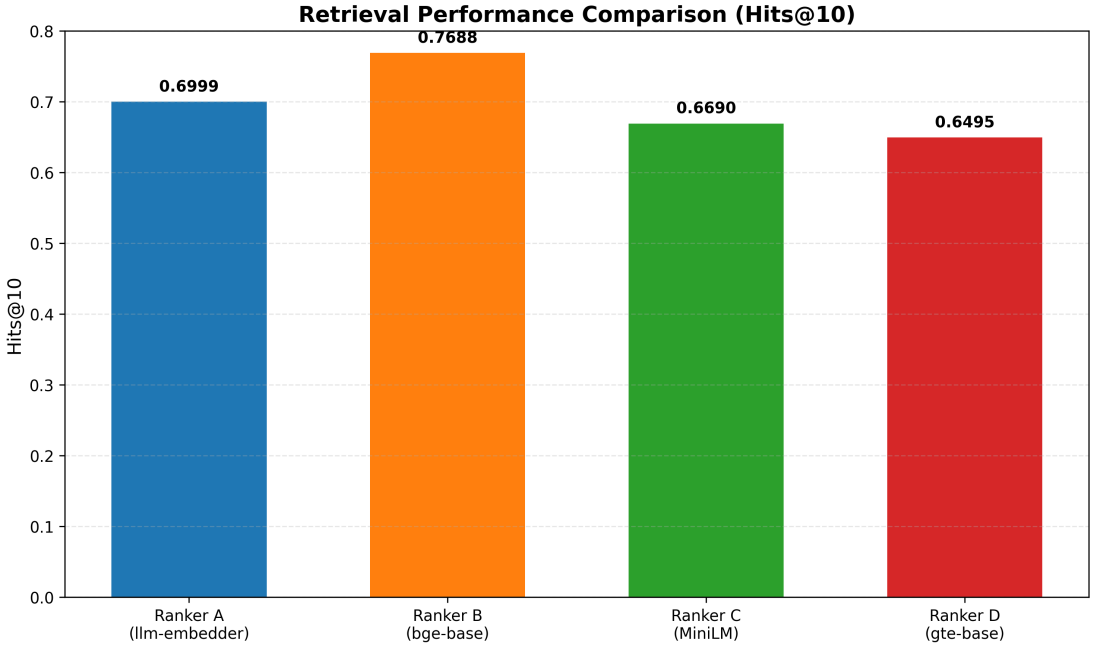

Figure 1: Retrieval Performance Comparison

provided multi-hop QA corpus, suggesting specialised training objectives do not always transfer to all task types.

5.2 Stage 2: RAG Performance

Overall results. Best overall system: Ranker B + Qwen2-7B (F1 = 0.42, Insufficient Info Accuracy = 75.4%).

The choice of LLM is the main contributing factor determining performance. Qwen2-7B reached F1 scores between 0.41 and 0.42 regardless of retriever used, while Llama-2-7b only achieved F1 = 0.23 across both retrievers. Retriever selection had a minimal effect – comparing Ranker B (76.88% retrieval rate) to Ranker A (69.99% retrieval rate) resulted in less than a 1% average F1 improvement, indicating that the reader stage is the primary bottleneck. In terms of handling insufficient information, Llama-2-7b performed very poorly (achieving just 0.7% accuracy on unanswerable queries), whereas Qwen2-7B correctly identified 75–77% of these cases, highlighting major differences in model calibration.

Component analysis. Retriever effect: comparing identical LLMs across retrievers reveals surprisingly minimal impact:

The 10% retrieval quality gap (Ranker B: 76.88% vs Ranker A: 69.99%) translated to <1% final F1 difference, indicating retrieval quality saturates (beyond ∼70% Hits@10), LLM limitations dominate end-to-end performance.

LLM effect: averaging across both retrievers shows clear LLM ranking:

Qwen2-7B outperformed all LLMs by >25% relative improvement over next-best (Llama-3). Llama-2-7b demonstrated consistent failure, likely due to older architecture (2022 knowledge cutoff, no GQA, shorter context).

The heatmap visualises interaction patterns across all 8 RAG configurations. The vertical striping pattern (consistent colors within each LLM column) confirms LLM choice is the dominant factor, while horizontal variations (retriever differences) are minimal.

Table 2: RAG System Performance (Full Results)

Retriever	LLM	Precision	Recall	F1	Insufficient Acc
Ranker A	Llama-2-7b	0.41	0.22	0.23	0.007
Ranker A	Llama-3-8B	0.33	0.32	0.32	0.419
Ranker A	Mistral-7B	0.42	0.31	0.32	0.568
Ranker A	Qwen2-7B	0.41	0.41	0.41	0.774
Ranker B	Llama-2-7b	0.40	0.22	0.23	0.007
Ranker B	Llama-3-8B	0.34	0.33	0.33	0.385
Ranker B	Mistral-7B	0.44	0.30	0.31	0.568
Ranker B	Qwen2-7B	0.42	0.42	0.42	0.754

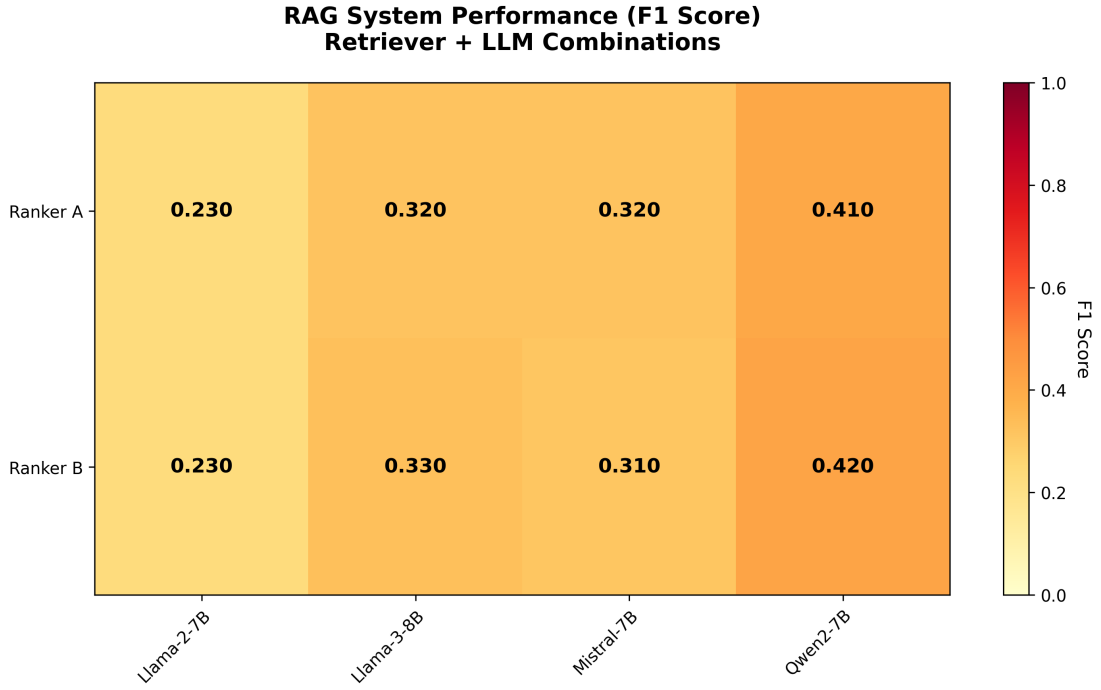

Figure 2: F1 Score Heatmap (Retriever × LLM)

LLM	Ranker A F1	Ranker B F1	Δ
Llama-2-7b	0.23	0.23	0.00
Llama-3-8B	0.32	0.33	+0.01
Mistral-7B	0.32	0.31	-0.01
Qwen2-7B	0.41	0.42	+0.01

LLM	Mean F1	σ	Range
Qwen2-7B	0.4150	0.01	0.41–0.42
Llama-3-8B	0.3250	0.01	0.32–0.33
Mistral-7B	0.3150	0.01	0.31–0.32
Llama-2-7b	0.2300	0.00	0.23–0.23

5.3 Interaction Effects

The components are nearly orthogonal – the "best" retriever + "best" LLM produces the "best" system, but retriever quality contributes minimally compared to LLM quality. This suggests non-linear error propagation: improved retrieval helps only when the reader can effectively utilise the context, and all tested LLMs struggle equally with multi-hop reasoning regardless of input quality.

6 Analysis and Discussion

6.1 Failure Analysis

Failure analysis examined specific queries where systems diverged to identify systematic patterns beyond aggregate metrics. Analysis of 10 queries where Ranker B consistently outperformed Ranker A (average correctness: 1.00 vs 0.00–0.50) revealed three failure modes. First, queries requiring synthesis across multiple named sources (e.g., "Which tool reported by both X and Y...") consistently favoured Ranker B, which correctly retrieved "ChatGPT" while Ranker A retrieved the semantically close but incorrect "OpenAI" across all four LLMs.

Second, entity disambiguation queries where multiple candidates share similar descriptors showed Ranker B retrieving more discriminative

Table 3: Component Interactions

Metric	Value	Observation
Best Retriever (B)	76.88%	10% better than Ranker A
Best LLM (Qwen2)	F1 = 0.415	80% better than worst LLM (Llama-2)
Best System (B+Qwen2)	F1 = 0.42	Matches best LLM performance
Retriever Improvement	+0.78%	Minimal impact on final F1

documents – correctly identifying "Sam Altman" as the Silicon Valley figure accused of fraud, while Ranker A retrieved wrong context leading to incorrect entities. Third, temporal reasoning queries with constraints like "following X event" required retrieving documents with explicit temporal relationships, where Ranker B's 10% Hits@10 advantage proved critical. Notably, failure analysis found zero queries where Ranker A consistently outperformed Ranker B, suggesting Ranker B's superiority is systematic across question types rather than complementary. The llm-embedder's "LLM-compatible" training objective showed no identifiable advantages over bge-base's general-purpose approach.

LLM behavior analysis examined 10 queries with maximal disagreement across models to identify failure modes. Llama-2-7b exhibited problematic calibration failure, achieving 0.7% accuracy on unanswerable queries (2/301 correct). Rather than responding "Insufficient Information," it consistently hallucinated answers. This overconfident generation led to high precision (0.40–0.41) but extremely low recall (0.22), likely resultant from its 2022 knowledge cutoff and lack of advanced calibration techniques. Llama-3-8B showed moderate improvement with 41.9% insufficient information accuracy, demonstrating that architectural advances (GQA, 8k context, 15T training tokens) improve calibration. However, it still missed 58% of unanswerable queries and showed inconsistency, preferring to generate answers over admitting uncertainty. Mistral-7B achieved 56.8% insufficient information accuracy and F1 = 0.31–0.32, proving more willing to abstain than Llama-3. However, it exhibited high variance across retriever contexts, with identical queries producing different answers depending on document ordering, suggesting weaker multi-hop integration. Qwen2-7B demonstrated superior performance with 75–77% insufficient information accuracy and highest F1 (0.41–0.42). Its larger context length (131k tokens), SwiGLU activation, and multilingual training enabled better calibration – correctly identifying unanswerable queries while maintaining high answerable accuracy (Qwen Team, 2024). Qwen2 also produced more contextual answers rather than bare entities, suggesting better instruction-following.

Of the 2,556 queries, 301 (11.8%) have gold label "Insufficient Information" – queries where the corpus lacks necessary evidence. Correctly identifying these is critical for production RAG systems to avoid hallucinations. Table 4 shows performance across all eight systems.

The results reveal several critical patterns. Llama-2 is production-unsafe with a 99.3% hallucination rate, fabricating answers for nearly every unanswerable query. Llama-3 improved to 41.9% accuracy (58× better), demonstrating that architectural advances significantly improve calibration. The 77% performance span (0.7% to 77.4%) between worst and best models shows that calibration ability is model-specific rather than purely dependent on architecture or training scale. Qwen2's multilingual training and longer context may encourage more conservative generation. Notably, retriever selection has minimal effect – within the same LLM, different retrievers yield similar insufficient information accuracy (e.g., ragA4: 77.4% vs ragB4: 75.4%), confirming that calibration is an LLM property rather than dependent on input quality.

6.2 Variance and Consistency

Per-query F1 distributions reveal that Qwen2 models exhibit the lowest variance ($\sigma = 0.01$) across retrievers, indicating robust performance independent of retrieval quality. In contrast, Llama-2 exhibits zero variance ($\sigma = 0.00$) due to consistent failure across all configurations – neither retriever could overcome its fundamental calibration issues. For production deployment, consistent mediocrity (Mistral: F1 = 0.31–0.32, $\sigma = 0.01$) may be preferable to high-variance systems where performance depends heavily on retrieval quality. However, Qwen2 achieves both high mean (F1 = 0.42) and low variance, making it optimal for reliability.

Table 4: Insufficient Information Performance by System

System	Correct "Insuf. Info"	Accuracy	Hallucination Risk
ragA1	2 / 301	0.7%	99.3% (worst)
ragB1	2 / 301	0.7%	99.3% (worst)
ragA2	126 / 301	41.9%	58.1%
ragB2	116 / 301	38.5%	61.5%
ragA3	171 / 301	56.8%	43.2%
ragB3	171 / 301	56.8%	43.2%
ragA4	233 / 301	77.4%	22.6% (best)
ragB4	227 / 301	75.4%	24.6%

6.3 Experimental Insights

Correct response parsing proved critical for evaluation validity. Each LLM uses different chat templates and special tokens: Llama-2/Mistral use instruction delimiters, Llama-3 uses structured role headers, and Qwen2 uses assistant role markers with end-of-turn terminators. Initial testing without proper parsing produced multi-paragraph responses instead of concise answers, rendering evaluation invalid. The `STAGING=True` iterative testing phase on sample queries was essential for identifying correct delimiters through manual inspection of raw model outputs, demonstrating that model-specific parsing logic is necessary for production RAG systems.

The minimal retriever effect (Ranker B +10% retrieval → +0.78% F1 improvement) reveals a retrieval quality ceiling beyond approximately 70% Hits@10, where LLM limitations dominate. If retrieval achieved 100% Hits@10 (perfect recall), the maximum possible F1 would still be bounded by LLM multi-hop reasoning capability. Results suggest this ceiling is around F1 = 0.42–0.45, given that Qwen2 + Ranker B (76.88% retrieval) achieved F1 = 0.42, and improving retrieval by 10% yielded only 1% F1 gain. Extrapolating, even perfect retrieval might add only 3–4% more F1. The pipeline exhibits non-linear error cascading – retrieval failures are fatal (no relevant documents makes the task impossible), but retrieval success only enables potential reader success. The reader can still fail with perfect input through hallucination, entity confusion, or poor multi-hop integration.

6.4 Lessons Learned

Several experimental design choices proved effective. Testing across model families (Llama, Mistral, Qwen) revealed that calibration and instruction-following vary independently of parameter count – 7B Qwen2 outperformed 8B Llama-3 by 28%. Pre-cached models on the GPU server avoided downloads and saved a lot of time and resources. Systematic failure analysis through random sampling of divergent queries (10 where Ranker B outperformed A, 10 with maximal LLM disagreement) revealed actionable patterns that aggregate metrics obscured, such as temporal reasoning as Ranker B's strength and calibration as Qwen2's advantage.

Moreover, several assumptions proved incorrect. First, the expectation that better retrieval guarantees better RAG performance was violated. Ranker B's 10% retrieval advantage translated to less than 1% F1 improvement, rather than the anticipated 5–10% gain. This outcome confirmed the non-linear component interactions described in Section 2. Second, the assumption that llm-embedder's "LLM-compatible" training objective would excel with LLMs found no support. Ranker A showed no query types where it outperformed the general-purpose Ranker B. This suggests that specialised training objectives may not transfer across all task types. Third, the initial prompt design overly constrained Qwen2's naturally contextual responses. Relaxing the prompt to "Answer directly without explanation" struck a better balance between conciseness and completeness.

These negative results demonstrate that theory does not always align with practice in Information Retrieval. Component-level optimisation does not guarantee system-level gains, and production RAG systems require end-to-end evaluation across component combinations – testing retrievers and LLMs in isolation is insufficient for predicting real-world performance.

7 Conclusion

This study evaluated Retrieval Augmented Generation systems for multi-hop question answering, testing 4 retrieval models and 8 retriever-LLM combinations (2 retrievers × 4 LLMs) on a 2,556-query dataset. Key findings address three research ques-

tions:

1. Ranker B (BAAI/bge-base-en-v1.5) achieved the best retrieval performance, yielding 76.88% Hits@10 – a 10% improvement over the next-best retriever. It worked especially well on more complex questions that required information from different sources or careful matching of names and dates.

2. Qwen2-7B-Instruct achieved F1 = 0.42, outperforming Llama-3-8B (0.33), Mistral-7B (0.32), and Llama-2-7b (0.23). The key difference was calibration – Qwen2 correctly identified 75–77% of unanswerable queries, while Llama-2 achieved only 0.7% and hallucinated answers instead. Parameter count proved less important than model training approach and calibration ability.

3. The best retriever combined with the best LLM produced the best system (F1 = 0.42), but retriever quality contributed less than 1% to final F1 despite 10% retrieval improvement. This shows that retrieval quality plateaus around 70% Hits@10 – beyond this threshold, LLM limitations in reasoning and calibration become the main bottleneck.

For RAG system design, these findings suggest prioritising LLM selection over retrieval optimisation. Once retrieval quality exceeds 70%, improving the LLM (through calibration) provides better gains than further retrieval improvements. This suggests allocating 70–80% of optimisation effort to the reader stage and 20–30% to retrieval. Calibration is critical for production systems – Llama-2's 99.3% hallucination rate on unanswerable queries makes it unsuitable for deployment regardless of its accuracy on answerable questions. RAG systems should be evaluated on both answerable and unanswerable queries, not just answerable F1 scores. Full pipeline evaluation is necessary because retrieval metrics like Hits@10 poorly predict final system performance. Ranker B's 10% retrieval advantage only delivered a 1% F1 improvement, showing that testing components separately does not actually predict real-world performance.

Several limitations affect how widely these findings apply. The 609-document corpus is small-scale, and the 70% retrieval saturation may not hold for production systems with millions of documents where baseline retrieval would be lower. GPU quota constraints limited testing to 7–8B parameter models; larger models (70B+) with stronger reasoning might show different patterns. The experiments used fixed temperature (0.1), top-k (10), and prompt format without testing alternatives, so different settings might reveal instances where retrieval quality becomes a more important factor in performance. The corpus was heavily skewed toward sports (35%) and technology (28%), and results may differ for cross-domain corpora with more diverse text types. Finally, exact-match evaluation penalises semantically correct but differently worded responses, potentially underestimating system performance on ambiguous queries.

Future work should test whether reranking (e.g., cross-encoders) can improve lower-performing retrievers to match Ranker B, checking if specialised retrieval methods can compensate for weaker embedding models. Testing retrieval quality at different levels would measure the saturation threshold more precisely and show whether the 70% plateau holds across different datasets and LLM families. Developing better metrics for model calibration on unanswerable queries could guide improvements through prompt engineering, fine-tuning, or confidence thresholds given the large 77% performance gap between models.

References

[1] Jurafsky, D., & Martin, J. H. (2024). Speech and Language Processing (3rd ed. draft). Chapter 14: Question Answering, Information Retrieval, and Retrieval-Augmented Generation. Retrieved from https://web.stanford.edu/~jurafsky/slp3/

[2] Li, Z., Zhang, X., Zhang, Y., Long, D., Xie, P., & Zhang, M. (2023). Towards general text embeddings with multi-stage contrastive learning. arXiv preprint arXiv:2308.03281.

[3] LlamaIndex Documentation. (2024). https://docs.llamaindex.ai/

[4] Meta AI. (2024). Llama 3 Model Card. https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md

[5] Mistral AI. (2024). Mistral-7B-Instruct-v0.3 Model Card. https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3

[6] Qwen Team. (2024). Qwen2 Technical Report. Alibaba Cloud. https://huggingface.co/Qwen/Qwen2-7B-Instruct

[7] Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). arXiv preprint arXiv:1908.10084.

[8] Touvron, H., Martin, L., Stone, K., et al. (2023). Llama 2: Open Foundation and Fine-Tuned Chat Models. Meta AI. https://huggingface.co/meta-llama/Llama-2-7b-chat-hf

[9] Xiao, S., Liu, Z., Zhang, P., & Muennighoff, N. (2023). C-Pack: Packaged Resources To Advance General Chinese Embedding. arXiv preprint arXiv:2309.07597. https://huggingface.co/BAAI/bge-base-en-v1.5

[10] Zhang, P., Xiao, S., Liu, Z., Dou, Z., & Nie, J. Y. (2023). Retrieve Anything To Augment Large Language Models. arXiv preprint arXiv:2310.07554.