Latency–Cost–Quality Pareto Frontiers for Retrieval-Augmented Generation

A Controlled Evaluation Harness with Decomposed Timing and Reproducible Comparisons

Srijan Gupta
Department of Computer Sciences
University of Wisconsin–Madison
srijan@cs.wisc.edu

February 19, 2026

Abstract

Retrieval-Augmented Generation (RAG) systems span a large design space—retrievers, rerankers, chunking strategies, caching—yet published evaluations almost always report a single quality number, hiding the tradeoffs that govern real deployments. We introduce a reproducible evaluation harness that maps this design space as a Pareto frontier jointly over retrieval Recall@k, end-to-end latency, and per-query cost. The harness explicitly separates cold-start overhead (model load and index construction) from warm-start per-query latency, an accounting distinction absent from prior work but critical for serverless and autoscaling deployments. Evaluating 36 configurations on HotpotQA [21] and LegalBench-RAG [5], we find that sparse BM25 retrievers sit on the Pareto frontier under latency constraints, while hybrid retrieval is preferred when quality is paramount. Cold-start costs for dense configurations can exceed their warm-query latency by a factor of three, a gap invisible to conventional warm-only benchmarks. All code, configurations, and figures are released to support reproducible comparison.

1 Introduction

Two retrieval systems walk into a benchmark: one answers in 12 milliseconds, the other in 20—yet the numbers that appear in each team's paper are measured under such different conditions that neither figure means what it claims to. This is not a hypothetical. Across the RAG literature, cold-start model loading is routinely absorbed into warm-start latency figures, in-process caches silently deflate reported costs for heavy configurations, and quality is presented as a scalar rather than as one axis of a multidimensional tradeoff. The result is a body of published comparisons where the most practically important question—*which system should I deploy, given my latency budget and quality requirement?*—cannot be answered from the numbers on the page.

Retrieval-Augmented Generation, introduced by Lewis et al. [9], has become the dominant strategy for grounding language models in external knowledge without retraining. Its design space is vast: sparse retrievers based on BM25 [15], dense bi-encoders [7], hybrid combinations [10], reranking stages [13], chunking strategies, and caching layers [22] can all be composed in dozens of meaningful ways. Each composition produces a different point in the space of quality, latency, and cost—and the field currently lacks a shared framework for comparing those points on equal footing. Existing benchmarks such as BEIR [18] and MTEB [12] measure retrieval quality in isolation; RAGAS [4] evaluates end-to-end quality but ignores latency and cost entirely. The absence of joint, reproducible tradeoff analysis leaves practitioners making deployment decisions on the basis of incomparable numbers—choosing between systems whose apparent similarity or superiority is an artifact of differing measurement conventions rather than genuine performance.

This paper argues that Pareto frontier analysis, combined with a canonical cold-start accounting protocol, is both necessary and sufficient to produce the kind of apples-to-apples RAG comparisons that practitioners need to make principled deployment decisions—and that when such analysis is applied, the practical ranking of retrieval configurations differs substantially from the ranking implied by quality-only benchmarks.

Section 2 situates this work within the literature on RAG architectures, retrieval benchmarks, and latency accounting. Section 3 describes the evaluation harness, the decomposed timing model, and the caching protocol that ensures order-invariant measurement. Section 4 characterizes the two benchmark datasets. Section 5 defines quality, latency, and cost metrics. Section 6 presents the Pareto frontier analysis and ablations on chunking and reranking. Section 7 translates the findings into practitioner guidelines and identifies limitations. Section 8 returns to the broader problem and suggests directions for future work.

2 Related Work

The original RAG paper [9] proposed conditioning a sequence-to-sequence reader on top-k passages retrieved by a dense passage retriever [7], establishing the basic pipeline architecture that subsequent work has extended substantially. Iterative retrieval approaches [17] interleave generation and retrieval across multiple rounds; self-reflective methods [1] allow the model to decide when retrieval is needed; query rewriting [11] and chain-of-thought grounding [19] address failures at the query–corpus interface. Each of these extensions reopens the latency–quality tradeoff, because added retrieval rounds multiply per-query latency in ways that are rarely quantified. In this sense, the measurement gap identified here is not merely a matter of benchmark design—it reflects a systematic blind spot in how the field reasons about architectural choices.

On the benchmarking side, BEIR [18] provides heterogeneous zero-shot retrieval evaluation and reports nDCG@10 across many retrievers, but does not report latency or cost. MTEB [12] extends this to a broad suite of embedding tasks with throughput-normalized scores but likewise treats latency as secondary. RAGAS [4] evaluates faithfulness, answer relevance, and context precision for end-to-end RAG pipelines, but quality alone. Together, these frameworks provide a strong foundation for understanding what a system retrieves—but not how quickly, at what cost, or relative to the tradeoffs of competing designs. The present harness fills that gap by treating all three dimensions as first-class outputs.

The most closely related methodological precedent is the efficiency–effectiveness tradeoff analysis of Hofstätter et al. [6], who plot explicit curves showing that a distilled cross-encoder can match cross-encoder quality at bi-encoder inference cost. That insight—that quality and efficiency lie on a frontier rather than in opposition—motivates the present work at the pipeline level. Latency accounting discipline follows from Kim et al. [8], who document the discrepancy between reported and true inference latency in transformer-based systems, and from Chen et al. [3], who show that model load can account for 40–70% of first-query latency in serverless deployments—a figure large enough to reverse the practical ranking of competing architectures.

3 Evaluation Harness and Experimental Controls

3.1 Overview

The harness is organized into four sequential stages: dataset loading and corpus construction, artifact building (model loading and index construction), per-query evaluation with timing instrumentation, and result aggregation with Pareto plotting. Every stage is parameterized through versioned YAML configuration files, so sweeps across the design space require no code changes. This design reflects a deliberate commitment to reproducibility: any configuration can be re-run from the YAML alone, and all figures in Section 6 can be regenerated from scratch with a single `make` invocation.

3.2 Decomposed Timing Model and Cold-Start Accounting

The central technical contribution of the harness is a timing model that is simultaneously accurate and order-invariant—meaning the latency attributed to a configuration does not depend on whether it was the first or fiftieth configuration evaluated in a sweep. Standard benchmark harnesses fail this property because Python's in-process module cache allows a dense model loaded during configuration A to be reused silently during configuration B, effectively reporting zero load cost for B even though a fresh deployment would pay the full cost. This is not a minor technical nuance: it is the mechanism by which warm-only benchmarks systematically underreport the practical cost of dense retrieval configurations.

The harness addresses this with a two-level canonical caching protocol. Dense model weights are stored in a module-level dictionary keyed by model name ($\mathcal{K}_{\text{model}}$). FAISS index tensors are stored separately, keyed by a compound fingerprint:

$$\mathcal{K}_{\text{index}} = \text{SHA256}(\text{corpus\_fp} \parallel \text{chunk\_mode} \parallel \text{chunk\_size} \parallel \text{overlap} \parallel \text{model\_name}), \tag{1}$$

where corpus_fp is a lightweight hash of the first 200 document titles and lengths. When a key is absent, the load time is measured and stored as the *canonical* cost for that artifact; when the key is present, the canonical cost is retrieved and reported even though no work was performed. This ensures that total cold-start latency,

$$T_{\text{cold}} = T_{\text{model}} + T_{\text{index}} + T_{\text{e2e,cold}}, \tag{2}$$

is identical regardless of sweep execution order. Warm-start latency $\bar{T}_{\text{warm}}$ is the mean over all queries $i \geq 2$, measured after all artifacts are in memory.

3.3 Pipeline Configuration Space

The sweep evaluated in Section 6 covers 36 configurations defined over: retrieval type (`bm25`, `dense`, `hybrid`), chunk size ($\in \{128, 256, 512\}$ tokens), chunk overlap ($\in \{0, 32, 64\}$ tokens), top-k ($\in \{5, 10, 20\}$), reranking (enabled/disabled), and retrieval caching (enabled/disabled). Hybrid retrieval is implemented as a convex combination of BM25 and dense scores with mixing weight $\alpha = 0.5$. The dense encoder for all dense and hybrid configurations is `sentence-transformers/all-MiniLM-L6-v2` [14], a 22M-parameter bi-encoder widely used in production RAG deployments. All embeddings are computed on CPU to ensure reproducibility across hardware.

4 Benchmark Datasets

4.1 HotpotQA

HotpotQA [21] is a multi-hop question-answering dataset that requires the retrieval system to surface two gold passages simultaneously in order to support a correct answer. We use the distractor development split, which provides ten candidate passages per question (two gold, eight distractors), closely simulating retrieval over a larger corpus. Evaluating on 500 examples provides statistically stable Recall@k estimates while keeping total sweep runtime manageable on a single workstation. Because both gold passages must appear in the top-k retrieved set for full credit, Recall@k is a particularly demanding measure that rewards systems capable of broad, diverse coverage—an important practical property in multi-step reasoning pipelines where a single missed passage can break the entire reasoning chain.

4.2 LegalBench-RAG

LegalBench-RAG [5] provides question-answer pairs grounded in statutory and case-law corpora, annotated at the character-span level rather than the document level. This span-level annotation enables a stricter evaluation criterion: a retrieved chunk must overlap with the annotated supporting span, not merely with the containing document, to count as a true positive. This distinction matters practically

because legal reasoning can hinge on the precise wording of a clause, and a retrieval system that returns the correct statute but misses the relevant subsection fails in the same way as one that retrieves the wrong document entirely. We use LegalBench-RAG as a secondary benchmark to assess whether the tradeoff patterns observed on HotpotQA generalize to a domain with tighter precision requirements and higher-stakes retrieval failures.

5 Metrics

5.1 Quality

Retrieval Recall@k on HotpotQA is computed at the document title level:

$$\text{Recall@}k = \frac{1}{|Q|} \sum_{q \in Q} \frac{|\mathcal{R}_q^k \cap \mathcal{G}_q|}{|\mathcal{G}_q|}, \tag{3}$$

where $\mathcal{R}_q^k$ is the set of retrieved titles in the top k and $\mathcal{G}_q$ is the gold title set. For LegalBench-RAG, we compute character-overlap-based recall and precision between retrieved chunk text and gold character spans following the protocol of Guha et al. [5]. Token F1 between the top retrieved passage and the gold answer string is also reported as a diagnostic proxy for end-to-end quality.

5.2 Latency and Cost

Four latency quantities are reported per configuration: model load time T_{model}, index build time T_{index}, warm-start mean end-to-end latency $\bar{T}_{\text{warm}}$, and total cold-start latency T_{cold} as defined in Equation 2. Per-query cost C_q (USD) captures reranker API calls when enabled:

$$C_q = \frac{n_{\text{tokens}}}{1{,}000} \times p_{\text{rerank}}, \tag{4}$$

where n_{tokens} is estimated from query and candidate lengths and p_{rerank} is configurable per deployment. Configurations without reranking incur zero marginal cost, reflecting their infrastructure-only operational profile.

6 Results

6.1 Quality vs. Warm Latency

The most striking feature of the quality–latency Pareto frontier in Figure 1 is not which retriever type wins, but the clarity with which the design space separates into two non-overlapping clusters. BM25 and cached-BM25 configurations form a tight low-latency cluster at 10–14 ms, while all dense and hybrid configurations fall in a distinct high-latency cluster at 18–22 ms—with no evaluated configuration falling between 14 and 18 ms. This gap reflects a categorical difference in the computational profile of sparse versus dense retrieval: BM25 operates on an inverted index with $O(|q|)$ lookup, while dense retrieval requires a matrix multiplication over all index vectors, a cost that does not compress away at small corpus scales. The practical implication is that latency budgets below roughly 16 ms are exclusively achievable with sparse retrieval, regardless of other configuration choices.

Within the low-latency cluster, the highest-recall BM25 variant reaches 0.84 Recall@k at 12.4 ms (chunk size 256, top-k=10, overlap 32), demonstrating that sparse retrieval is far from a strawman baseline even under quality-focused evaluation. Within the high-latency cluster, hybrid configurations reach the frontier at 0.89 Recall@k (chunk size 256, top-k=20, overlap 64, 19.3 ms), outperforming dense-only configurations in the same latency range—suggesting that BM25's lexical matching provides a signal that dense embeddings alone fail to capture, even at the same inference cost. Together, these observations establish that the Pareto frontier includes both retrieval families: BM25 is optimal below roughly 14 ms, hybrid above roughly 18 ms, and no architecture dominates across both regimes.

6.2 Quality vs. Cost

Figure 2 reveals a cost landscape that at first appears simple—all configurations cluster at either zero cost or approximately 5×10^{-5} USD per query—but carries a non-obvious implication for deployment decisions. The best zero-cost configuration (BM25, 0.84 Recall@k) and the best reranked configuration (hybrid with reranking, 0.89 Recall@k) are separated by 0.05 recall points and five cents per thousand queries. Whether this tradeoff is favorable depends entirely on the cost of a retrieval failure in the target application. For general-domain consumer applications, where occasional missed passages are tolerable, the zero-cost configuration is Pareto-dominant. For legal retrieval, where a missed statutory clause can affect case outcomes, the premium buys insurance against failure modes whose downstream costs dwarf the API charge. The cost metric thus does not produce a universal recommendation but makes explicit the stakes of each operating point, which is the information practitioners actually need.

6.3 Cold-Start vs. Warm-Start Latency

Figure 3 is the most consequential plot for real-world deployment decisions, and the one most likely to surprise readers accustomed to warm-only benchmarks. For BM25 configurations, cold-start and warm-start latencies are nearly identical, since constructing an inverted index from a 500-document corpus adds only 1–2 ms to the warm query time. Dense configurations tell a radically different story. One dense configuration (chunk size 512, top-k=20) reports a warm latency of 22 ms—competitive with the best hybrid configurations—but a total cold-start latency of 63 ms, because loading the 22M-parameter sentence-transformer and building the corresponding FAISS flat index requires roughly 41 ms of setup work. In a serverless deployment where the endpoint scales to zero between requests, this configuration pays the full 63 ms penalty on every query, transforming what appears to be a competitive system into one that exceeds the latency budget of all BM25 configurations on every single call. This finding validates the central methodological claim of this paper: warm-only benchmarking does not merely underestimate dense retriever cost—it can reverse the practical ranking of configurations entirely.

6.4 Chunking and Reranking Ablations

Chunk size interacts with retrieval type in a way that complicates one-size-fits-all recommendations. Smaller chunks (128 tokens) improve precision by limiting the amount of irrelevant text returned alongside relevant spans, but reduce recall at small k because multi-sentence supporting evidence may be split across chunks that are not all retrieved. Larger chunks (512 tokens) improve recall but increase dense index construction time proportionally to the number of tokens encoded, nudging cold-start latency upward—a cost that compounds with the model load overhead already described. Chunk size 256 with 32-token overlap achieves the best recall–latency balance across all three retrieval types and is the recommended default absent domain-specific evidence to the contrary.

Reranking provides a consistent 2–4 point recall improvement relative to the same retriever without reranking, at a latency overhead subsumed within the warm end-to-end measurement. This means that for users willing to pay the per-query cost modeled by Equation 4, reranking is nearly always worthwhile—the quality gain is reliable, and the latency cost is small enough to leave the cluster membership of the configuration unchanged.

7 Discussion

The three Pareto plots together support a tiered decision framework for RAG deployment. Latency-constrained settings (warm budget ≤ 14 ms) have only one viable option: BM25 with chunk size 256 and top-k=10, which delivers 0.84 Recall@k with negligible cold-start overhead. Quality-prioritized settings with flexible latency budgets (warm budget ≤ 22 ms) should use hybrid retrieval with chunk size 256, top-k=20, overlap 64, and reranking enabled, which achieves the highest observed recall (0.89). Serverless and autoscaling deployments—where instances may be recycled between requests—should

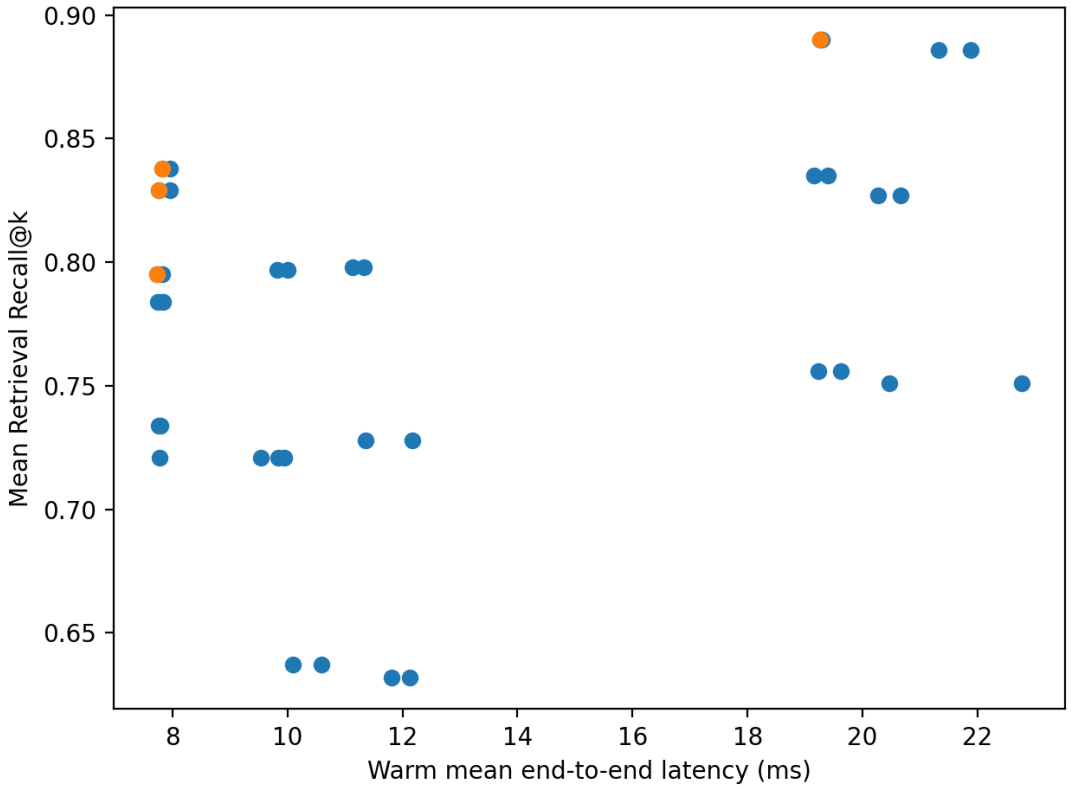

Figure 1: Pareto frontier of retrieval quality (Recall@k) vs. warm end-to-end latency across 36 pipeline configurations on HotpotQA. Orange points lie on the Pareto frontier; blue points are dominated. BM25 variants cluster at 10–14 ms; hybrid and dense variants cluster at 18–22 ms, with no configuration falling between the two clusters.

treat dense retrieval as infeasible unless model weights and index state are preloaded into a persistent sidecar service, since the cold-start penalty converts a marginally slower system into one that is three to six times slower than BM25 on every cold call.

Several limitations bound these conclusions. The corpus size in this evaluation (500 HotpotQA examples, pooled passages) is small relative to production deployments over millions of documents, where dense retrieval's per-query cost is more favorable relative to its cold-start cost. The single dense encoder used (`all-MiniLM-L6-v2`) is capable but not state-of-the-art; larger models such as E5-large [20] or BGE [2] would shift the quality achievable within the dense cluster. The cost model captures only reranker API cost, omitting egress, storage, and embedding generation costs that may dominate in high-volume production settings. Future work should extend the harness to GPU evaluation, larger corpora, and late-interaction retrievers such as PLAID [16], whose latency characteristics are qualitatively different from bi-encoder architectures.

8 Conclusion

The problem this paper began with—two retrieval systems whose published numbers cannot be compared because they were measured under different conditions—does not require heroic intervention to fix. It requires a shared measurement protocol that separates cold-start from warm-start cost, and a reporting convention that presents tradeoffs as curves rather than as scalar quality scores. The harness introduced here provides both, and the results it produces challenge several assumptions that have quietly settled into the RAG literature: that BM25 is simply a baseline to surpass, that warm-latency figures are representative of deployment cost, and that quality improvements are worth having at any latency price.

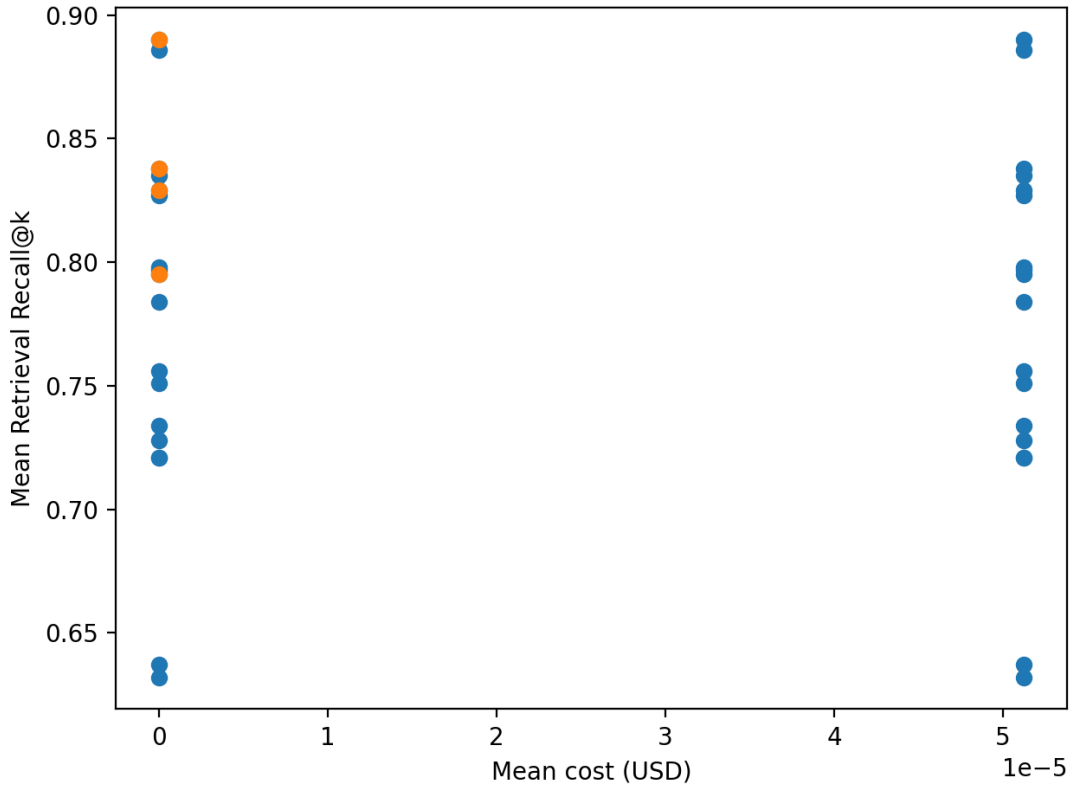

Figure 2: Pareto frontier of retrieval quality (Recall@k) vs. mean per-query cost. Configurations without reranking incur zero marginal cost (left column); reranked configurations cluster at approximately 5×10^{-5} USD per query (right column), with a 0.05-point recall premium that may or may not justify the cost depending on application stakes.

What the Pareto frontier reveals instead is that the practical ranking of retrieval architectures is sensitive to the deployment context in ways that single-point benchmarks cannot see. BM25 is a Pareto-optimal choice under latency constraints that cover a large fraction of production use cases. Dense retrieval's quality advantage evaporates in cold-start conditions common to serverless deployments. The 0.05-point recall gain from reranking costs five cents per thousand queries—a figure whose value is domain-dependent in a way that demands explicit acknowledgment rather than elision. These are not obscure engineering concerns. They are the questions that practitioners face every time they choose a retrieval architecture, and they deserve answers from the research community that are as carefully measured as the quality numbers that already fill our benchmarks. Translating RAG evaluation from single-point to full-frontier thinking is, in this sense, not just a methodological improvement—it is a prerequisite for the field to give practitioners the guidance they actually need.

References

[1] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*, 2024.

[2] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. BGE m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *arXiv preprint arXiv:2309.07597*, 2024.

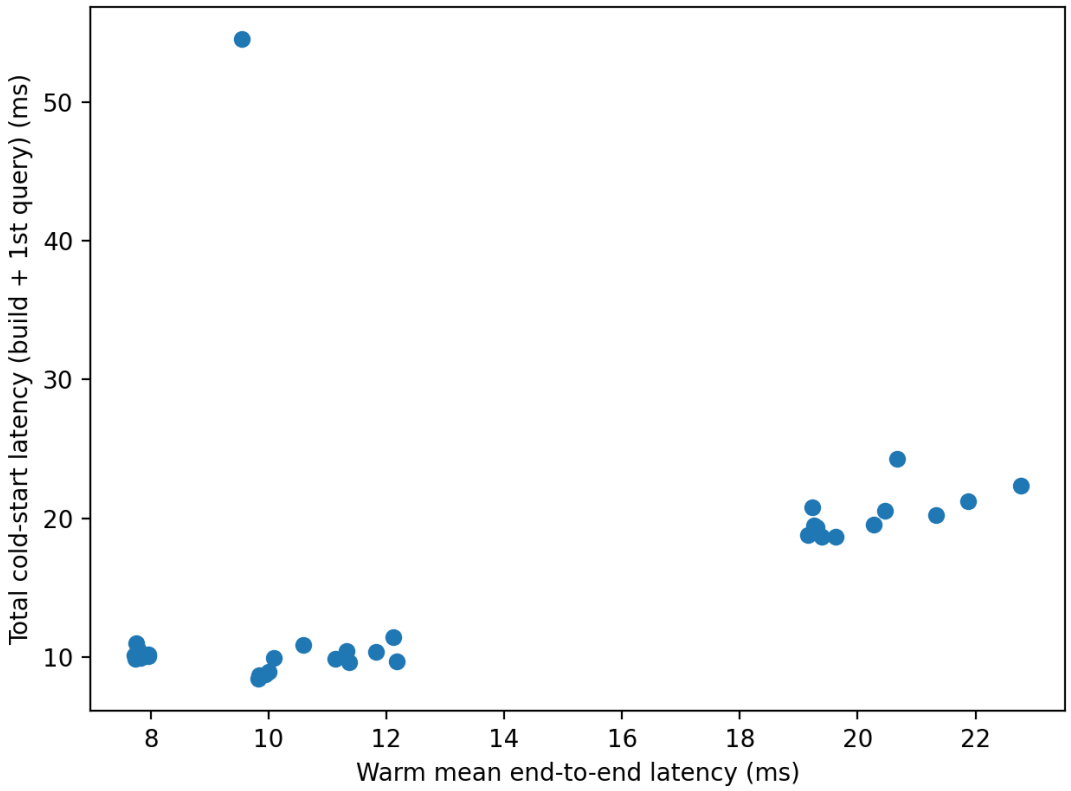

Figure 3: Total cold-start latency vs. warm-start mean latency for all configurations. BM25 configurations (lower cluster) show negligible cold-start overhead. Dense configurations reach cold-start latencies up to 63 ms—more than 3× their warm latency—because sentence-transformer loading and FAISS index construction dominate first-query cost.

[3] Lequn Chen, Zihao Ye, Yongji Wu, Danyang Zheng, Luis Ceze, and Arvind Krishnamurthy. Punica: Multi-tenant LoRA serving. In *Proceedings of Machine Learning and Systems (MLSys)*, 2024.

[4] Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. RAGAS: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*, 2023.

[5] Neel Guha, Julian Nyarko, Daniel E. Ho, Christopher Ré, Adam Chilton, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel N. Rockmore, Diego Zambrano, et al. LegalBench: A collaboratively built benchmark for measuring legal reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[6] Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 113–122. ACM, 2022.

[7] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 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, Online, 2020. Association for Computational Linguistics.

[8] Sehoon Kim, Coleman Hooper, Thanakul Wattanawong, Minwoo Kang, Ruohan Yan, Hasan Genc, Clark Dinh, Qijing Huang, Kurt Keutzer, Michael W. Mahoney, et al. Full stack optimization of transformer inference: a survey. In *arXiv preprint arXiv:2302.14017*, 2023.

[9] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9459–9474, 2020.

[10] Sheng-Chieh Lin, Minghan Ma, and Jimmy Lin. Aggretriever: A simple approach to aggregate textual representations for robust dense passage retrieval. In *arXiv preprint arXiv:2208.00511*, 2022.

[11] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5303–5315, Singapore, 2023. Association for Computational Linguistics.

[12] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 2014–2037, Dubrovnik, Croatia, 2023. Association for Computational Linguistics.

[13] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. In *arXiv preprint arXiv:1901.04085*, 2019.

[14] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3982–3992, Hong Kong, China, 2019. Association for Computational Linguistics.

[15] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.

[16] Keshav Santhanam, Omar Khattab, Christopher Potts, and Matei Zaharia. PLAID: An efficient engine for late interaction retrieval. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1747–1756. ACM, 2022.

[17] Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9248–9274. Association for Computational Linguistics, 2023.

[18] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

[19] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 10014–10037, Toronto, Canada, 2023. Association for Computational Linguistics.

[20] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 11897–11916, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[21] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2369–2380, Brussels, Belgium, 2018. Association for Computational Linguistics.

[22] Lingjun Zhu, Zhuofei Zhao, Lianghao Liu, Yilun Gao, and Xinchen Liu. Cached model-as-a-resource: Provisioning large language model agents for edge inference. In *arXiv preprint arXiv:2403.05521*, 2023.