Designing Retrieval Systems for LLMs: A Controlled Comparison of Chunk-Based and File-Centric Methods

Siddhanth Kalyanaraman[1,*], Ravi Gupta[2], Ashish Sandeep Joshi[1], Kaushik Sandeep Joshi[1], Jayesh Suryawanshi[1], and Vidney Jadhav[1]
* *Corresponding Author: siddhank@bu.edu*

[1] *Boston University* [2] *AI researcher*

***Abstract*—Retrieval Augmented Generation (RAG) has become a widely used approach for enabling large language models (LLMs) to access proprietary or private data, but its pipelines require complex infrastructure including chunking, embedding, and vector database management. Structured Sequential File Retrieval (SSFR), in which documents are ranked at the file level using lexical scoring and the best snippets extracted as context, offers reduced complexity and latency but lacks empirical comparison with RAG under controlled conditions. This study introduces a decision framework for selecting retrieval architectures by comparing RAG with SSFR across two open-weight models (LLaMA-3.3-70B-Instruct, Qwen2.5-32B-Instruct), two quantization levels, and two corpus sizes. A benchmark of 69 prompts spanning factual, analytical, and multi-hop reasoning queries is evaluated using quality metrics (hallucination rate, groundedness, answer correctness) and operational metrics (latency, GPU utilization, throughput, cost). Results show that SSFR matches RAG on small to medium corpora while reducing latency and cost, but structured chunk-level retrieval provides stronger grounding as corpus size and query complexity increase. The study produces a decision matrix linking workload characteristics to recommended configurations.**

***Keywords*—retrieval-augmented generation, document question answering, large language models, hallucination detection, quantization, decision framework**

I. Introduction

Large language models (LLMs) have advanced natural language understanding and generation but remain limited by reliance on pretraining data and susceptibility to hallucination—producing fluent text unsupported by source evidence [1], [2]. Retrieval-augmented approaches mitigate this by grounding generation in externally retrieved documents, improving factual fidelity for knowledge-intensive tasks [3].

The dominant paradigm, *Retrieval-Augmented Generation* (RAG), segments source documents into fixed-size chunks, encodes them as dense embeddings, and retrieves semantically relevant passages as generation context [3], [4]. While effective, RAG introduces considerable system complexity: document preprocessing, text chunking, embedding generation, vector database management, and multi-stage retrieval increase operational overhead and deployment cost.

An alternative approach which we formalize as *Structured Sequential File Retrieval* (SSFR) operates at the whole-file level: documents are ranked using composite lexical scoring, and the best snippets from top-ranked files are extracted as context. SSFR eliminates the need for vector databases and embedding pipelines, reducing both computational overhead and system complexity. Recent improvements in LLM reasoning and long-context processing suggest that such simplified strategies may suffice for many practical use cases.

Despite extensive research on RAG components [4]–[6], no controlled study has compared chunk-level against whole-file retrieval under matched model, quantization, and corpus conditions. Existing evaluations assess RAG in isolation [7] or compare within the same paradigm [8]. This gap is consequential: practitioners must rely on untested assumptions about how architectural choices interact with model capability, precision, and corpus characteristics.

This paper makes four contributions:

1) A controlled factorial experiment comparing RAG and SSFR across multiple configurations, using matched models (LLaMA-3.3-70B-Instruct and Qwen2.5-32B-Instruct), evaluation protocols, and hardware.
2) An LLM-as-judge evaluation protocol with claim-level grounding verification, relevance scoring, and per-query GPU telemetry, evaluated on a benchmark of 69 prompts spanning factual, analytical, and multi-hop reasoning queries.
3) Identification of paradoxical findings including a confidence-hallucination inversion and a quantization non-effect that challenge common assumptions about the relationship between model capability, retrieval complexity, and output quality.
4) A weighted multi-criteria decision framework that adaptively routes queries to the optimal retrieval architecture based on query complexity, model capacity, and corpus characteristics, producing a decision matrix that links workload characteristics to recommended system configurations.

II. Related Work

Retrieval-Augmented Generation. Lewis et al. [3] introduced RAG as a framework combining Dense Passage Retrieval (DPR) [9] with a sequence-to-sequence generator.

Subsequent work has refined chunking strategies [10], reranking, and self-reflective retrieval [5]. Gao et al. [4] survey modular RAG architectures, identifying retrieval, augmentation, and generation as independent design axes. Our work varies the retrieval axis while holding augmentation and generation constant.

Dense vs. Sparse Retrieval. Dense retrieval using bi-encoders [9], [11] achieves strong semantic matching but can miss lexical signals. Hybrid approaches combining dense and sparse scoring [12], [13] improve robustness. BM25 [14] remains competitive for exact-match and keyword-heavy queries. Our RAG pipeline uses a hybrid dense-lexical reranker (80/20 weighting), while SSFR employs purely lexical scoring at the file level.

Hierarchical and Whole-File Retrieval. Standard RAG at the chunk level can fragment context. SSFR-style systems perform deterministic file ranking followed by intra-document snippet extraction, preserving document structure and incorporating adaptive context handling (grounded, partial, or no-context modes) to control generation based on retrieval strength.

Evaluation of Grounded Generation. RAGAS [7] uses LLM judges for faithfulness and relevance evaluation. DeepEval [15] extends this with claim-level assessment. Zheng et al. [16] validated LLM-as-judge reliability. Our protocol adopts claim-level grounding with three-label classification.

Quantization for LLM Inference. Dettmers et al. [17] showed 4-bit NF4 quantization preserves quality; Frantar et al. [18] and Lin et al. [19] demonstrated similar results. We extend quantization analysis to retrieval-augmented contexts.

Long-Context and Document-Level Retrieval. Gemini [20] and recent context-extension work [21] enable processing of entire documents within the context window. Anthropic [22] demonstrated 100K+ token context handling. SSFR represents a structured approach to document-level retrieval that does not require extended context windows, using snippet extraction to manage token budgets.

Decision Frameworks and Model Capability. Advances in reasoning-capable LLMs suggest that stronger models may tolerate incomplete retrieval contexts, potentially reducing retrieval complexity requirements. Adaptive routing between models or strategies has been explored for mixture-of-experts [23] and cascading inference [24]. Our decision framework extends this concept to retrieval architecture selection, using empirically calibrated quality–operational tradeoff weights.

III. System Architecture

We implement two retrieval pipelines sharing a common LLM backend, evaluation protocol, and prompt design (Fig. 1). Both follow a retrieval $\rightarrow$ context construction $\rightarrow$ generation pipeline but differ in retrieval granularity. RAG performs chunk-level semantic retrieval with hybrid reranking; SSFR adopts hierarchical whole-file ranking with targeted snippet extraction, preserving document structure and mitigating chunk-level fragmentation.

A. RAG Pipeline

Source documents (PDF and TXT) are loaded via LangChain [10] and segmented into 700-character chunks with 180-character overlap using a recursive character splitter respecting paragraph and sentence boundaries. Chunks are encoded using the `all-MiniLM-L6-v2` sentence transformer [11] (384 dimensions, cosine similarity) and indexed in ChromaDB [25].

At query time, a *hybrid balanced retriever* retrieves a candidate pool of 60 chunks via dense similarity, re-scores each using dense (w=0.80) and lexical overlap (w=0.20) [12], balances across source files (max 3/file), and expands neighbors ($\pm$1). Context strength is estimated via retrieval scores and query coverage, routing to *grounded* (top $\geq$0.50, coverage $\geq$0.30), *partial* (top $\geq$0.30 or coverage $\geq$0.18), or *no-context* prompts. Complex queries receive expanded context (14K vs. 9K chars) and increased top-k.

B. Structured Sequential File Retrieval Pipeline

A file-level catalog indexes all corpus files with metadata and summaries. For each query, every file is scored using a composite lexical metric:

$$S_{\text{file}} = \max\left(S_{\text{full}},\ 1.15 \cdot S_{\text{simp}},\ 1.25 \cdot S_{\text{key}}\right) \tag{1}$$

where S_{full}, S_{simp}, and S_{key} are scores computed on the full query, a simplified (stopword-removed) query, and a leading keyphrase (top 8 tokens), respectively. Each individual score combines four sub-metrics: exact phrase matching ($\times$1000 bonus), token overlap ratio, token frequency density, and ordered token span matching. Question-bank files are penalized ($\times$0.08) to prevent answer leakage.

The top-4 files are selected, and from each, the top-5 highest-scoring snippets (1800 characters, 350-character overlap) are extracted with neighbor inclusion. Context status thresholds are calibrated to the lexical score range: *grounded* requires file score $\geq$12.0 and top snippet $\geq$6.5. A *refusal rescue* mechanism detects hedging phrases in the generated answer and re-prompts with a general-knowledge template when triggered.

C. Shared Infrastructure

Both pipelines use an identical `LLMManager` wrapping HuggingFace Transformers [26] with BitsAndBytes [17] quantization (4-bit NF4 with double quantization, or 8-bit), attention optimization (flash attention [27] for RAG, SDPA for SSFR), and configurable GPU memory bounds. Generation uses low-temperature decoding (T=0.2, no sampling, max 384 tokens). The same LLM serves as both generator and judge, ensuring prompt-template fairness across pipelines.

IV. Evaluation Protocol

Claim-Level Grounding. Each answer is segmented into sentences (up to 12 claims). Each claim is judged against retrieved context as *supported* (1.0), *partially supported* (0.5), or *unsupported* (0.0). Groundedness score is the mean claim score; hallucination rate is the fraction of unsupported claims.

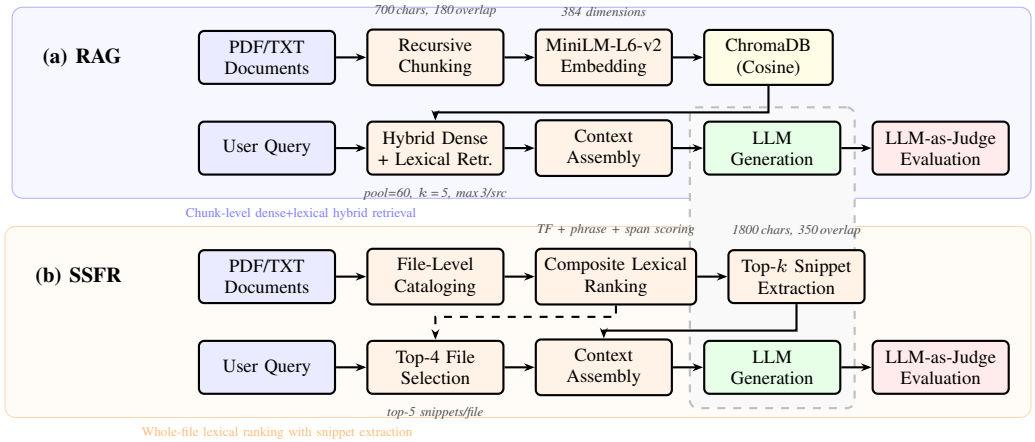

Fig. 1. Dual-pipeline architecture. **(a) RAG**: documents are chunked (700 chars, 180 overlap), embedded via MiniLM-L6-v2 [11] into 384-dim vectors, indexed in ChromaDB (cosine similarity), and retrieved using a hybrid dense-lexical reranker (candidate pool of 60, k=5, balanced across sources with max 3 chunks per file). **(b) SSFR**: files are cataloged and ranked using composite lexical scoring (token overlap, frequency, exact phrase match, ordered span), top-4 files selected, and top-5 snippets per file (1800 chars, 350 overlap) extracted. Both pipelines share the same LLM backend (HuggingFace Transformers with BitsAndBytes quantization) and LLM-as-judge evaluation protocol.

Together, these metrics quantify answer correctness—the degree to which generated content is factually grounded in retrieved evidence. This follows the DeepEval [15] and RAGAS [7] protocols.

Relevance Scoring. Answer and context relevance are scored on 1–5 Likert scales, separating retrieval quality from generation quality.

GPU Telemetry. A threaded NVML monitor (50ms polling) records per-query average/peak GPU utilization, memory consumption, PyTorch peak allocated memory, and sample count. Token throughput, effective throughput (throughput $\times$ utilization), and end-to-end latency are logged.

Statistical Methods. Significance is assessed via Welch's t-test (two-tailed, unequal variance); effect sizes are Cohen's d ($|d|$<0.2: negligible, 0.2–0.5: small, 0.5–0.8: medium, >0.8: large). Significance levels: *p<0.05, **p<0.01, ***p<0.001.

V. Experimental Design

A full 2^4 factorial design crosses four independent variables (Table I). All experiments run on a single NVIDIA H200 (80 GB HBM3) GPU and A40 GPU. The *small/medium* corpus comprises history and nationalism texts; the *big* corpus comprises religious and philosophical texts. Each configuration is evaluated on a 69-prompt benchmark (39 small-corpus + 30 big-corpus queries) spanning factual, analytical, and multi-hop reasoning categories, all requiring multi-sentence structured answers. The system dynamically adjusts retrieval for complex prompts (increased top-k, expanded context windows). Metrics are aggregated per-query with paired comparisons across both systems, yielding 582 total observations.

A. Model Selection and Quantization

LLaMA-3.3-70B-Instruct represents a high-capacity reasoning model; Qwen2.5-32B-Instruct serves as a resource-efficient alternative. This pairing tests whether stronger reasoning compensates for simpler retrieval. Both use 4-bit and 8-bit quantization with GPU-aware memory allocation, enabling 70B+ models on single-GPU hardware.

TABLE I
Experimental Design: 2^4 Full Factorial

Variable	Levels	Values
Retrieval Approach	2	RAG, SSFR
Base Model	2	Qwen2.5-32B-Instruct (32B), LLaMA-3.3-70B-Instruct (70B)
Quantization	2	4-bit NF4 (double quant.), 8-bit
Corpus Size	2	Small/medium (history), Big (philosophy)

VI. Results

A. RAG vs. SSFR: Head-to-Head

Table II presents the primary comparison. RAG achieves substantially lower hallucination (−21.4 pp, d=−0.76) and higher groundedness (+20.2 pp, d=0.74), both medium-to-large effects. RAG also retrieves more relevant context (3.76 vs. 2.58, d=1.10, large) and responds 33% faster (39.7s vs. 59.7s, d=−0.78). However, SSFR achieves significantly higher answer relevance (4.73 vs. 4.48, d=−0.48), indicating that whole-file context enables more complete answers. The core trade-off is factual correctness (RAG) versus answer completeness (SSFR).

B. Model Reasoning Capability and Retrieval Interaction

Model. LLaMA-70B outperforms Qwen-32B on all quality metrics: lower hallucination (−6.4 pp, p<0.01, d=0.21), higher groundedness (+13.3 pp, p<0.001, d=−0.47), and substantially higher answer relevance (4.83 vs. 4.42, p<0.001, d=−0.82, large). LLaMA requires 3$\times$ peak GPU memory

TABLE II
RAG VS. SSFR: QUALITY, RETRIEVAL, AND PERFORMANCE

Metric	RAG	SSFR	d	Sig.
Answer Quality				
Hallucination	0.195	0.409	−0.76	***
Groundedness	0.685	0.484	0.74	***
Answer Rel.	4.48	4.73	−0.48	***
Context Rel.	3.76	2.58	1.10	***
Retrieval Characteristics				
Docs Retrieved	7.59	1.49	—	***
Query Coverage	0.49	1.00	−4.14	***
Confidence	0.541	0.991	−6.63	***
Performance				
Response (s)	39.7	59.7	−0.78	***
Throughput (t/s)	10.6	10.0	0.13	n.s.
GPU Mem (%)	57.1	63.0	−0.20	*

n_{RAG}=276, n_{SSFR}=306. Cohen's d reported; Welch's t-test.
Sig.: *** $p<0.001$, ** $p<0.01$, * $p<0.05$, n.s. = not significant.

TABLE III
QUANTIZATION IMPACT: QUALITY PRESERVED, EFFICIENCY DOUBLED

Metric	4-bit	8-bit	d	Sig.
Hallucination	0.324	0.293	0.10	n.s.
Groundedness	0.573	0.585	−0.04	n.s.
Answer Rel.	4.66	4.58	0.15	n.s.
Throughput (t/s)	14.27	6.71	2.29	***
Eff. Throughput	11.70	4.28	3.18	***
Peak Mem (MB)	35,664	40,776	−0.22	**
Cost (USD)	1.73	2.02	−0.61	***

Welch's t-test. *** $p<0.001$, ** $p<0.01$, n.s. = not significant.

(58,962 vs. 19,761 MB, d=−2.85), presenting a clear quality–resource tradeoff. Importantly, the larger model shows greater tolerance to incomplete retrieval contexts: when paired with SSFR, LLaMA-70B achieves higher groundedness than Qwen-32B paired with RAG in several configurations, suggesting that model reasoning capability can partially compensate for simpler retrieval methods.

Quantization. The most practically significant null finding: 4-bit quantization produces *no* significant quality change (hallucination p=0.22, groundedness p=0.61, relevance p=0.07) while delivering 2.13× throughput (14.27 vs. 6.71 tok/s, $p<0.001$, d=2.29), 12.5% memory savings, and 14.5% cost reduction (Table III).

C. Corpus Size Effects

The larger corpus produces better quality: lower hallucination (−13.7 pp, $p<0.001$, d=0.47), higher groundedness (+10.0 pp, $p<0.001$), and higher context relevance (3.56 vs. 2.78, d=−0.67). Answer relevance shows no difference (p=0.80). For small/medium corpora, SSFR often matches RAG while reducing latency; as corpus size and complexity increase, RAG provides stronger grounding.

TABLE IV
OPTIMAL CONFIGURATIONS BY DEPLOYMENT PRIORITY

Priority	Configuration	Value
Min. Halluc.	RAG | Qwen-32B | 8-bit | Big	0.086
Max. Ground.	RAG | LLaMA-70B | 4-bit | Big	0.817
Min. Latency	RAG | Qwen-32B | 4-bit | Big	16.0 s
Max. Relev.	RAG | LLaMA-70B | 4-bit | Big	5.00
Best SSFR Rel.	SSFR | LLaMA-70B | 4-bit | Big	4.90
Min. Memory	RAG | Qwen-32B | 4-bit | Small/medium	25.5 GB

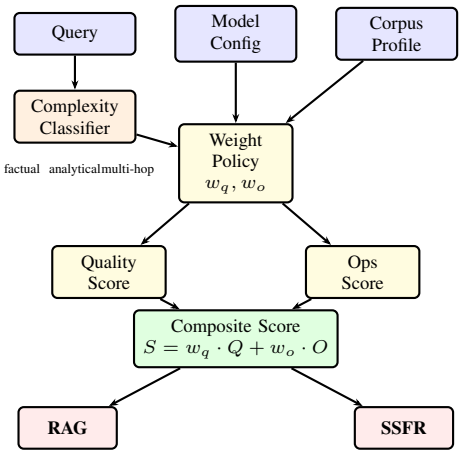

Fig. 2. Adaptive decision framework. Query complexity is classified (factual, analytical, or multi-hop), combined with model capacity and corpus profile to produce context-dependent weights w_q (quality) and w_o (operational). RAG and SSFR are scored on composite quality (hallucination, groundedness, relevance) and operational (latency, throughput, utilization) metrics, then routed to the higher-scoring architecture.

D. Best Configurations by Priority

Table IV reports optimal configurations per priority. While SSFR has higher mean relevance (4.73 vs. 4.48), the best individual configuration is RAG with LLaMA-70B at 5.00 indicates the larger model's reasoning overcomes RAG's narrower context.

VII. DECISION FRAMEWORK

The results demonstrate that no single architecture dominates across all conditions. We therefore develop a *weighted multi-criteria decision framework* (Fig. 2) that routes queries to the optimal retrieval architecture, producing a decision matrix that links workload characteristics to recommended system configurations.

A. Query Complexity Classification

Queries are classified into three categories using keyword-based heuristics: *factual* (direct knowledge retrieval), *analytical* (evaluation, significance, impact), and *multi-hop* (comparison, contrast, hypothetical reasoning). Classification drives the quality–operational weight balance.

B. Scoring Functions

For each (model, corpus, complexity) configuration, quality and operational scores are computed:

$$Q = w_h \cdot \hat{H} + w_g \cdot \hat{G} + w_a \cdot \hat{A}_r + w_c \cdot \hat{C}_r \tag{2}$$

$$O = 0.48 \cdot \hat{L} + 0.34 \cdot \hat{U} + 0.18 \cdot \hat{T} \tag{3}$$

where $\hat{\cdot}$ denotes min-max normalized metrics (inverted for lower-is-better), H is hallucination rate, G is groundedness, A_r/C_r are answer/context relevance, L is latency, U is GPU utilization, and T is throughput. Quality weights (w_h, w_g, w_a, w_c) are complexity-dependent: multi-hop queries weight hallucination at 0.35, and factual/analytical at 0.32.

C. Adaptive Weighting

The final routing score is $S = w_q \cdot Q + w_o \cdot O$, where w_q and w_o are determined by query complexity, corpus size, and model capacity. Base weights start at w_q=0.42, w_o=0.58, then shift by complexity (e.g., multi-hop: w_q=0.56; factual: w_q=0.24), corpus (big: w_q+0.08; small/medium: w_q−0.04), and model size (large: w_q−0.04; medium: w_q+0.03).

D. Conservative Decision Policy

Beyond raw score comparison, a set of empirically calibrated guard-rails prevent edge-case misrouting. For example, RAG is preferred only when the quality gap exceeds complexity-dependent thresholds (e.g., hallucination gap $\geq$0.12 and grounding gap $\geq$0.08 for factual queries on large corpora). Ties and near-ties (score difference $<$0.06) default to SSFR, which has lower infrastructure overhead (no vector database).

VIII. Discussion

A. Paradoxical Findings

Confidence–Hallucination Inversion. SSFR reports near-perfect confidence (0.991) yet hallucinates at 2× the rate of RAG (0.409 vs. 0.195). Across all 582 observations, Pearson $r = +0.320$ between confidence and hallucination, meaning *higher confidence predicts more hallucination*. This has immediate implications for production systems: confidence-based filtering is counterproductive in SSFR architectures. The inversion arises because SSFR's confidence is derived from file-level lexical scores (S_{file}), which saturate when any query tokens appear in the document, regardless of whether the model faithfully uses that content.

Context–Grounding Paradox. SSFR achieves 98% grounded-context status and 100% query coverage, yet produces lower groundedness scores. This reveals that *access to full source documents does not guarantee faithful utilization*. We hypothesize that larger context windows induce the "lost in the middle" phenomenon [28], where models attend preferentially to context boundaries and generate beyond the evidence.

TABLE V
Key Metric Correlations (n=582)

	Ground.	**Confid.**	**Resp. Time**
Hallucination	−0.902	+0.320	+0.047
Groundedness	—	−0.254	−0.157
Answer Rel.	−0.044	+0.220	+0.017

General Knowledge as Quality Signal. RAG falls back to general knowledge 31.9% of the time (vs. SSFR's 2.0%), yet achieves better groundedness. This suggests that RAG's adaptive degradation mixing partial retrieval with background knowledge outperforms SSFR's all-or-nothing delivery. This finding supports hybrid retrieval-generation strategies [5].

Quantization Non-Effect. Unlike model choice, corpus, and retrieval architecture, quantization shows *no significant effect on any quality metric*. This is a practically valuable null finding: 4-bit quantization achieves 2.13× throughput with negligible quality cost, strongly supporting aggressive quantization for deployment. This extends the findings of Dettmers et al. [17] from fine-tuning to retrieval-augmented inference.

B. Architectural Implications

The hallucination–relevance tradeoff is the central finding: RAG provides better grounding fidelity while SSFR provides more complete answers. The larger LLaMA-70B shows greater tolerance to incomplete retrieval than Qwen-32B, suggesting that as model capability improves, the threshold for requiring complex retrieval shifts upward particularly for small/medium corpora. However, for complex multi-hop queries on large corpora, structured retrieval remains essential regardless of model size. A productive future direction is a *hybrid architecture* using RAG's chunked retrieval for evidence-critical passages and SSFR's file-level context for completeness, routed by the decision framework.

The correlation matrix (Table V) reveals strong coupling between hallucination and groundedness (r=−0.90), validating their complementarity as evaluation metrics. Confidence shows weak positive correlation with hallucination (r=+0.32), confirming the inversion finding. Response time shows near-zero correlation with quality metrics ($|r|$<0.16), indicating that faster responses do not sacrifice quality within these architectures.

Retrieval design can matter more than model size or context size alone; hybrid systems are likely to be optimal in deployment.

IX. Conclusion and Future Work

We present a controlled factorial comparison of chunk-level RAG and whole-file SSFR across 582 observations

and 16 configurations. RAG achieves 52% lower hallucination and 42% higher groundedness while being 33% faster; SSFR yields 5.3% higher answer relevance. Model reasoning partially compensates for simpler retrieval on small/medium corpora, but structured retrieval remains essential for complex queries on large corpora. The decision framework provides context-sensitive routing producing a decision matrix linking workload characteristics to configurations. We recommend RAG for fidelity-critical applications and SSFR where completeness is paramount. The quantization non-effect 4-bit precision doubling throughput with no quality cost has immediate deployment value.

Future work includes extending to larger models on multi-GPU hardware, evaluating diffusion-based LLMs [30], expanding the benchmark, adopting independent judge models [29], and learning decision-framework weights from production telemetry.

REFERENCES

[1] L. Huang *et al.*, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *arXiv:2311.05232*, 2023.

[2] Z. Ji *et al.*, "Survey of hallucination in natural language generation," *ACM Comput. Surveys*, vol. 55, no. 12, pp. 1–38, 2023.

[3] P. Lewis *et al.*, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proc. NeurIPS*, 2020.

[4] Y. Gao *et al.*, "Retrieval-augmented generation for large language models: A survey," *arXiv:2312.10997*, 2024.

[5] A. Asai *et al.*, "Self-RAG: Learning to retrieve, generate, and critique through self-reflection," in *Proc. ICLR*, 2024.

[6] O. Yoran, T. Wolfson, O. Ram, and J. Berant, "Making retrieval-augmented language models robust to irrelevant context," in *Proc. ICLR*, 2024.

[7] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "RAGAS: Automated evaluation of retrieval augmented generation," in *Proc. EACL*, 2024.

[8] J. Chen *et al.*, "Benchmarking large language models in retrieval-augmented generation," in *Proc. AAAI*, 2024.

[9] V. Karpukhin *et al.*, "Dense passage retrieval for open-domain question answering," in *Proc. EMNLP*, 2020.

[10] H. Chase, "LangChain," 2023. [Online].

[11] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proc. EMNLP*, 2019.

[12] J. Chen *et al.*, "BGE M3-Embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation," *arXiv:2402.03216*, 2024.

[13] X. Ma *et al.*, "Fine-tuning LLaMA for multi-stage text retrieval," *arXiv:2310.08319*, 2023.

[14] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," *Found. Trends Inf. Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.

[15] Confident AI, "DeepEval: LLM evaluation framework," 2024. [Online].

[16] L. Zheng *et al.*, "Judging LLM-as-a-judge with MT-Bench and Chatbot Arena," in *Proc. NeurIPS*, 2023.

[17] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient finetuning of quantized language models," in *Proc. NeurIPS*, 2023.

[18] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate post-training quantization for generative pre-trained transformers," in *Proc. ICLR*, 2023.

[19] J. Lin, J. Tang, H. Tang, S. Yang, W.-M. Chen, W.-C. Wang, G. Xiao, X. Dang, C. Gan, and S. Han, "AWQ: Activation-aware weight quantization for LLM compression and acceleration," in *Proc. MLSys*, 2024.

[20] Gemini Team, Google, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," *arXiv:2403.05530*, 2024.

[21] Y. Chen *et al.*, "LongLoRA: Efficient fine-tuning of long-context large language models," in *Proc. ICLR*, 2024.

[22] Anthropic, "The Claude model family," Tech. Rep., 2024.

[23] A. Jiang *et al.*, "Mixtral of Experts," *arXiv:2401.04088*, 2024.

[24] L. Chen, M. Zaharia, and J. Zou, "FrugalGPT: How to use large language models while reducing cost and improving performance," *arXiv:2305.05176*, 2023.

[25] Chroma, "Chroma: The AI-native open-source embedding database," 2024. [Online].

[26] T. Wolf *et al.*, "HuggingFace's Transformers: State-of-the-art natural language processing," in *Proc. EMNLP: System Demonstrations*, 2020.

[27] T. Dao, "FlashAttention-2: Faster attention with better parallelism and work partitioning," in *Proc. ICLR*, 2024.

[28] N. F. Liu *et al.*, "Lost in the middle: How language models use long contexts," *Trans. ACL*, vol. 12, pp. 157–173, 2024.

[29] OpenAI, "GPT-4 technical report," *arXiv:2303.08774*, 2024.

[30] S. Nie *et al.*, "Large language diffusion models," *arXiv:2502.09992*, 2025.