Compact Language Models in Retrieval-Augmented Generation: Accuracy–Latency Trade-offs under a Unified Pipeline

Alexandros Papafragkakis[‡§], Yannis Tzitzikas[‡§]

[‡]Department of Computer Science, University of Crete, Greece
[§]Institute of Computer Science, Foundation for Research and Technology-Hellas, Greece

***Abstract*— Production deployment of Retrieval-Augmented Generation (RAG) systems requires balancing answer quality with computational constraints and response latency. While existing research has extensively investigated retrieval strategies and prompt engineering, the selection of the language model component remains relatively unexplored, particularly for resource-constrained scenarios where compact models (1B–8B parameters) offer practical advantages. This work systematically evaluates six compact open-source models within a controlled RAG pipeline across two complementary knowledge-intensive QA benchmarks. Our evaluation, using fixed retrieval configuration across nearly 16,500 test questions, reveals that parameter count alone is an unreliable predictor of RAG performance. Mid-size models approach or match larger counterparts in accuracy while offering substantial latency advantages, whereas smaller models benefit disproportionately from retrieval augmentation. Latency decomposition identifies distinct optimization priorities: retrieval accounts for most of the measured latency for smaller models, while generation becomes more prominent for larger ones. These findings suggest that strategic model selection can achieve near-optimal accuracy (within 1 percentage point of an 8B model) while reducing end-to-end latency from 0.80s to 0.61s in our evaluation under identical retrieval.**

I. Introduction

Large language models can generate fluent text but may produce unsupported statements when factual grounding is required. This occurs because model parameters are frozen post-training and knowledge is derived entirely from static training corpora. RAG addresses this limitation through a two-stage approach [1]: first retrieving relevant documents from local or external sources, then conditioning generation on these retrieved documents to ground responses in verifiable evidence.

However, most RAG research focuses on large-scale models (70B+ parameters) deployed on multi-GPU infrastructure. Such deployments are prohibitively expensive for resource-constrained organizations or mobile/edge applications. Several compact models have emerged recently, including Phi-3 Mini (3.8B), Gemma 2 (2B), and Llama 3.2 (1B and 3B variants). These can execute on single consumer GPUs with sub-second latencies and operational costs orders of magnitude lower than larger counterparts. Yet it remains unclear which compact models perform best in RAG contexts.

The literature provides limited guidance here. Existing studies typically evaluate single models in isolation, compare vastly different scales where conclusions do not transfer, or rely on proprietary APIs preventing reproducibility. We lack systematic understanding of how 2B vs 3B vs 7B models compare under the same retrieval setup, including retriever, reranker, and prompt template. Practitioners deploying RAG systems face this choice regularly but lack empirical data.

This paper addresses this gap through controlled evaluation of six compact models (Llama 3.2 1B, Gemma 2 2B, Llama 3.2 3B, Phi-3 Mini 3.8B, Mistral 7B, Llama 3.1 8B) in a controlled RAG pipeline with a fixed retrieval configuration and prompt template. We hold the retrieval and prompting components fixed (same embedder, hybrid search, reranker, and prompt template), so observed differences primarily reflect the generator model choice. Evaluation uses two complementary benchmarks: MetaQA [2] with 9,948 single-hop test questions requiring reasoning over movie data, and WC2014QA [3] with 6,483 single-hop test questions about 2014 FIFA World Cup requiring precise entity extraction. We measure both accuracy (Hits@1, F1, EM) and latency (retrieval, generation, end-to-end).

Results show model size poorly predicts RAG performance. On WC2014QA, Mistral 7B achieves 93.7% at 0.61s while Llama 3.1 8B reaches equivalent accuracy at 0.80s. On MetaQA, Llama 3.2 3B attains 87.8% (only 0.8pp below 8B) while maintaining sub-second latency. Notably, smaller models benefit disproportionately from RAG: Llama 3.2 1B improves 13× on MetaQA (5.5% to 72.5%), whereas Llama 3.1 8B improves 2.4× (36.2% to 88.6%). Latency analysis reveals retrieval dominates for smaller models (60-70% of runtime) while generation dominates for larger ones (40-50%), suggesting different optimization priorities.

Our contributions are fourfold: (1) systematic comparison of six compact LMs in a controlled RAG pipeline with fixed retrieval configuration, isolating model effects from retrieval/prompting/evaluation; (2) evaluation on two fundamentally different benchmarks capturing realistic query diversity; (3) joint accuracy-efficiency analysis with latency decomposition identifying system bottlenecks; (4) deployment-oriented insights showing Mistral 7B on efficiency frontier, compact models benefiting disproportionately from RAG, and retrieval optimization yielding higher returns than generation optimization for smaller models.

II. Background and Related Work

Our work builds on two research areas: retrieval-augmented generation and compact language model development. We review key concepts before positioning our contribution.

RAG Systems. Large language models generate text based solely on training data. When asked about recent publications or domain-specific facts, they either acknowledge limitations or fabricate details. Model parameters remain frozen post-training with no mechanism for incorporating new information. RAG systems split the task: retrieve relevant documents from local or external sources, then condition generation on retrieved documents [1]. Rather than relying on memorized patterns, the model extracts answers from provided reference text.

Standard RAG pipelines have three components. During indexing, documents get chunked, embedded via sentence transformers, and stored in vector databases alongside lexical indexes (BM25). At query time, retrieval searches for passages semantically similar to the question, often combining dense and sparse methods [4]. Generation formats passages as context, prepending them to questions before feeding to the LLM.

Most RAG research optimizes retrieval: better embeddings [5], hybrid ranking, multi-hop strategies for complex questions [6]. The generator receives less attention, treated as fixed. Researchers select a large model (GPT-3.5, Llama 2 70B), optimize retrieval parameters, evaluate the complete system. This leaves one question unexplored: does generator choice matter when high-quality context is provided?

Compact Language Models. The evaluated models represent different design philosophies. Mistral 7B [7] employs grouped-query attention and sliding-window attention for efficient 8K-token context handling, achieving performance rivaling models twice its size. Gemma 2 2B [8] uses local attention patterns and GeGLU activations to run on minimal hardware, showing strong reasoning despite 2B parameters. Phi-3 Mini [9] (3.8B) emphasizes high-quality training data over scale, focusing on structured reasoning relevant to RAG tasks. Llama 3.2 [10] offers 1B for edge deployment and 3B as middle ground, inheriting Llama 3 architectural improvements.

These span an order of magnitude (1B to 8B) but share constraints: consumer hardware, interactive latency, minimal memory. Whether differences translate to meaningful RAG performance gaps remains unclear.

Related Work. Lewis et al. [1] introduced RAG combining DPR with BART, demonstrating gains on Natural Questions [11] and TriviaQA [12]. Subsequent work expanded along multiple dimensions: contrastive learning for retrievers [4], iterative retrieval for multi-hop questions [6], hybrid dense-sparse methods [5].

Parallel research on compact models includes distillation approaches (DistilBERT [13], TinyBERT [14]) and data curation methods (Phi-2 [15], StableLM [16]). However, these focus on general benchmarks (MMLU [17], reasoning tasks) rather than RAG-specific evaluation.

Gap. No prior work systematically compares compact models in controlled RAG settings. Studies either compare single models against baselines [18], test vastly different scales where conclusions do not generalize [19], or use proprietary APIs preventing reproduction. We lack clear understanding of how 2B vs 3B vs 7B models perform under the same retrieval setup, including retriever, reranker, and prompt template.

III. Experimental Methodology

We design a controlled comparison isolating generator effects from retrieval quality.

Experimental Infrastructure. All experiments use the SemanticRAG pipeline [20] as experimental infrastructure, with no changes to retrieval or indexing components. We access the pipeline through its HTTP API endpoint, keeping the retrieval and indexing components fixed and only swapping the generator LLM parameter. This controlled setup ensures that observed performance differences stem solely from model selection rather than pipeline variations. Note that latency measurements include network overhead from HTTP communication.

Models. Six compact models are evaluated: Llama 3.2 1B, Gemma 2 2B, Llama 3.2 3B, Phi-3 Mini 3.8B, Mistral 7B, and Llama 3.1 8B. All models use the same retrieval configuration (embeddings, hybrid search, reranking), though retrieval executes independently per model. Performance differences reflect model characteristics given the same retrieval setup.

Retrieval Pipeline. The retrieval component uses dense embeddings (nomic-embed-text, 768d) combined with BM25 for hybrid search. Documents are chunked into 500-character segments with 50-character overlap. Per question, the system retrieves candidate passages via hybrid search and applies cross-encoder reranking (gte-reranker-modernbert-base). The reranked passages are provided as context for generation (system configuration targets top-10 passages; actual count may vary based on available results). Generation models receive context plus question, producing answers with temperature=0 for deterministic output.

Benchmarks. MetaQA [2] contains movie questions at three difficulty levels: single-hop ("Who directed Inception?"), two-hop ("Who acted in movies directed by Christopher Nolan?"), and three-hop requiring longer reasoning chains. We evaluate only on single-hop questions, using the test set of 9,948 questions. Multi-hop questions would require iterative retrieval and reasoning over multiple facts, confounding our goal of isolating generator performance under fixed retrieval context. The SemanticRAG pipeline employed in this study implements single-hop retrieval, which aligns with our objective of evaluating generator model performance given fixed, high-quality contextual input. The knowledge base consists of 43,234 movie entities derived from MetaQA triples [2], verbalized into natural language passages using the approach described in [20].

WC2014QA tests factual extraction about 2014 FIFA World Cup. Questions involve player statistics, match results, team rosters. We evaluate only on single-hop questions, using the test set of 6,483 questions with 1,236 supporting passages from news reports and match summaries.

These questions align with the single-hop retrieval capabilities of our experimental infrastructure and demand precise entity extraction where name errors constitute failure.

Metrics. Accuracy uses three metrics. Exact Match requires predicted answer matching ground truth exactly after normalization. Token-level F1 computes word overlap between prediction and reference. Hits@1 checks if answer matches any valid reference (useful for multiple correct answers). Latency decomposes into retrieval time, generation time, end-to-end time per query.

Procedure. For each model, we process all test questions. Retrieval configuration remains fixed across all models (same embeddings, same hybrid search parameters, same reranking model). Retrieval is executed independently per run under the same configuration for all models through the HTTP API; minor variations in retrieved passages may occur across runs due to backend state or serving effects. We record predictions and timing, compute metrics. Results represent aggregate performance across all test questions; given the large test sets (9,948 and 6,483 questions), absolute accuracy differences of 2-3 percentage points or more represent substantial performance distinctions, though formal statistical significance testing is left for future work. In practice, retrieval latency and aggregate retrieval behavior were largely stable across runs and models (Figure 2), indicating that observed performance differences are primarily attributable to the generator.

Implementation Details. Models are accessed through the SemanticRAG HTTP API [20]. Memory management, timeouts, and error handling are implemented in the evaluation client. Individual queries timeout at 60s to avoid stalled executions. Failed queries retry up to 3 times. Progress checkpoints periodically so interruptions do not lose work. The remote API uses 4-bit quantization (bitsandbytes NF4) for all models. Random seeds are fixed for reproducibility. To ensure fair comparison across models: identical retrieval configuration (same embedder, same hybrid search, same reranker, same top-k target), deterministic generation (temperature=0, max_new_tokens=128), and sequential single-query inference (batch size 1).

Latency Measurement Setup. All experiments were conducted on research computing infrastructure[1] using a single AMD Radeon RX 9070XT GPU (16GB VRAM). Models execute via remote HTTP API with 4-bit quantization. Generation uses greedy decoding (temperature=0) with max_new_tokens=128. Latency is measured using Python's perf_counter() capturing HTTP request-response round-trip time. We report service-level endpoint latencies measured at the client, which include network overhead and server-side processing (embedding computation, indexing operations, model inference, response formatting). Retrieval index loading time is excluded from per-query latency measurements. We report mean latency and median latency over test sets after 5 warm-up queries to stabilize system state and reduce sensitivity to outliers. Latency is decomposed into two service endpoints: retrieval service latency (covers embedding, hybrid search, and reranking) and generation service latency (covers model forward pass and decoding).

[1]Computational resources provided by the Institute of Computer Science, Foundation for Research and Technology – Hellas (ICS-FORTH).

Prompt Format and Normalization. Retrieved passages are formatted as a bulleted list under "Documents:" and provided to the model along with task instructions. The prompt template instructs the model to extract answers from documents, respond with only the exact entity name, and avoid null responses. The system message is: "Respond to all questions directly without any explanation." Models generate free-form text responses. For evaluation, we apply lightweight, dataset-specific normalization consistent with the dataset's entity formatting conventions. Predictions undergo: (1) right-strip whitespace, (2) replace spaces with underscores, (3) lowercase conversion, (4) truncate at the first comma to remove trailing qualifiers or enumerations, retaining the primary entity mention. This operation is applied only to model predictions and not to reference answers. All normalization steps are deterministic and applied consistently across models. Gold answers are provided in underscored format and are lowercased for comparison. This process handles the common case where models generate natural text (e.g., "Alan Pulido") while reference answers use underscored entities (e.g., "alan_pulido"). Exact Match requires identical normalized strings; F1 computes token overlap after normalization; Hits@1 checks if any reference answer matches the normalized prediction.

IV. Results

We report results across both benchmarks for all six models. Tables show accuracy metrics (Hits@1, F1, EM) for baseline (no RAG) and RAG-augmented configurations. Given the large test set sizes (9,948 and 6,483 questions), small absolute differences are unlikely due to sampling noise; we therefore focus on effect sizes (absolute percentage-point differences). Formal statistical significance testing would strengthen conclusions but is left for future work.

On MetaQA, Llama 3.1 8B achieves highest accuracy at 88.6% Hits@1. Llama 3.2 3B follows at 87.8% (0.8pp gap). Gemma 2 2B reaches 83.2%. The smallest model, Llama 3.2 1B, attains 72.5%. Phi-3 Mini and Mistral 7B score 80.6% and 82.7% respectively.

	Baseline			**RAG**		
Model	**H@1**	**F1**	**EM**	**H@1**	**F1**	**EM**
L3.2-1B	5.49	5.77	0.80	72.47	59.34	29.87
G2-2B	18.53	15.73	6.39	83.22	81.48	67.98
L3.2-3B	23.52	19.11	7.61	87.83	80.82	61.16
Phi-3	7.96	6.07	0.69	80.62	72.77	49.63
M-7B	31.83	26.62	8.01	82.65	80.21	**68.76**
L3.1-8B	**36.19**	**31.07**	**15.02**	**88.61**	**82.83**	65.42

Table 1: MetaQA: Baseline vs RAG performance. All metrics in %. H@1 = Hits@1, EM = Exact Match.

F1 and EM metrics follow similar patterns. Llama 3.1 8B leads at 82.8% F1 and 65.4% EM. EM scores are substantially lower than Hits@1 across all models, indicating models often produce approximately correct answers (matching on F1) without exact string equality.

On WC2014QA, the picture changes. Mistral 7B and

Llama 3.1 8B both reach 93.7% and 94.4% respectively, near-perfect on this factual extraction task. Phi-3 Mini achieves 81.2%. Llama 3.2 3B scores 76.8%.

Model	Baseline Hits@1 (%)	RAG Hits@1 (%)
L3.2-1B	6.03	66.68
G2-2B	12.90	68.98
L3.2-3B	13.11	76.78
Phi-3	11.34	81.19
M-7B	8.21	93.71
L3.1-8B	**20.66**	**94.42**

Table 2: WC2014QA: Baseline vs RAG performance.

Comparing baseline (no RAG) to RAG-augmented results reveals patterns. On MetaQA without retrieval, models perform poorly: Llama 3.2 1B at 5.5%, Llama 3.1 8B at 36.2%. With RAG, these jump to 72.5% and 88.6% (13× and 2.4× improvements respectively). The gap narrows as size increases: smaller models benefit disproportionately from retrieval, larger models already encode more factual knowledge.

V. Analysis

The raw results reveal patterns relevant to deployment decisions. We identify three key observations.

First, model size poorly predicts performance within the compact range. On MetaQA, Llama 3.2 3B (87.8%) nearly matches Llama 3.1 8B (88.6%) with less than half the parameters. Phi-3 Mini (3.8B) outperforms Mistral 7B on latency while trailing slightly on accuracy. This non-monotonic relationship indicates architectural choices, training data quality, and fine-tuning matter comparably to parameter count.

Second, bottlenecks depend on model size. For Phi-3 Mini and Mistral 7B, retrieval consumes 60-65% of total latency. Optimizing these requires faster embeddings, better indexing, or reduced passage counts. For Llama models, generation dominates at 50-55% runtime. Here, quantization, speculative decoding, or prompt compression provide better returns. One-size-fits-all optimization fails; deployment constraints should guide selection.

Third, task characteristics matter substantially. WC2014QA shows clear scaling benefits (93.7% for 7B-8B vs 66-77% for 1B-3B), while MetaQA compresses the range (72-89%). Factual extraction (matching exact entities from context) apparently requires more capacity than multi-hop reasoning when retrieval provides good passages. Applications dominated by entity lookup may need 7B+ models, while QA over documents can use 3B effectively.

On WC2014QA, Mistral 7B occupies the efficiency frontier: 93.7% accuracy at 0.61s latency. Llama 3.1 8B matches accuracy but adds 31% latency. Phi-3 Mini offers fastest responses (0.62s) at lower accuracy (81.2%). For deployments where both matter (production APIs with SLAs), Mistral 7B dominates.

Smaller models show surprising RAG gains. Llama 3.2 1B improves 13× on MetaQA with retrieval, Llama 3.1 8B gains 2.4×. This diminishing return pattern means if RAG infrastructure exists, 2-3B models become viable even if 70B performs better without RAG.

Cross-dataset comparison: MetaQA's multi-hop questions compress rankings, WC2014QA's factual extraction expands them. Standard deviation of Hits@1 is 5.8pp on MetaQA, 11.2pp on WC2014QA. When retrieval succeeds (providing all necessary facts), even small models chain reasoning steps.

EM vs Hits@1 gaps widen for smaller models. Llama 3.2 1B shows 42.6pp gap (72.5% Hits@1, 29.9% EM), Llama 3.1 8B has 23.2pp (88.6% vs 65.4%). Smaller models correctly identify answer concepts but struggle with formatting, normalization, exact phrasing.

VI. Conclusion

We evaluated six compact language models (1B–8B parameters) within a controlled Retrieval-Augmented Generation (RAG) pipeline across two question answering benchmarks, reporting accuracy metrics (Hits@1, token-level F1, Exact Match) alongside end-to-end latency with retrieval and generation breakdowns. Our results show that model size alone is a weak predictor of RAG performance. On MetaQA, Llama 3.2 3B approaches the accuracy of its 8B counterpart, while on WC2014QA, Mistral 7B achieves the most favorable accuracy–latency trade-off compared to Llama 3.1 8B.

These findings indicate that effective RAG deployment should prioritize system-level co-design and informed model selection under realistic latency and resource constraints, rather than relying solely on parameter scale. Table 3 summarizes practical deployment recommendations derived from our analysis.

Table 3: Deployment guidance in compact RAG settings.

Scenario	Model	Rationale
High-accuracy QA	Mistral 7B	Near-ceiling accuracy with lower latency than 8B models.
Entity-centric QA	Llama 3.2 3B	Accuracy comparable to 8B on MetaQA at substantially lower cost.
Strict latency budget	Phi-3 Mini	Fast generation when end-to-end latency dominates.
Cost-sensitive setups	Llama 3.2 1B–3B	Larger gains from improving retrieval than scaling the generator.

In production scenarios requiring high factual extraction accuracy, Mistral 7B lies on the accuracy–latency efficiency frontier. When strong retrieval is available, Llama 3.2 3B delivers near-optimal accuracy for entity-centric QA at significantly lower computational cost. Applications with stringent latency constraints may instead favor Phi-3 Mini, despite reduced accuracy.

A key takeaway is that the optimal model choice depends on which component dominates end-to-end latency. For compact models (1B–3B), retrieval accounts for the majority of runtime, suggesting that improvements in embeddings, caching, and index efficiency yield larger gains than modest increases in model size. Conversely, for larger models (7B–8B), generation becomes the dominant latency factor, making decoding and serving optimizations more im-

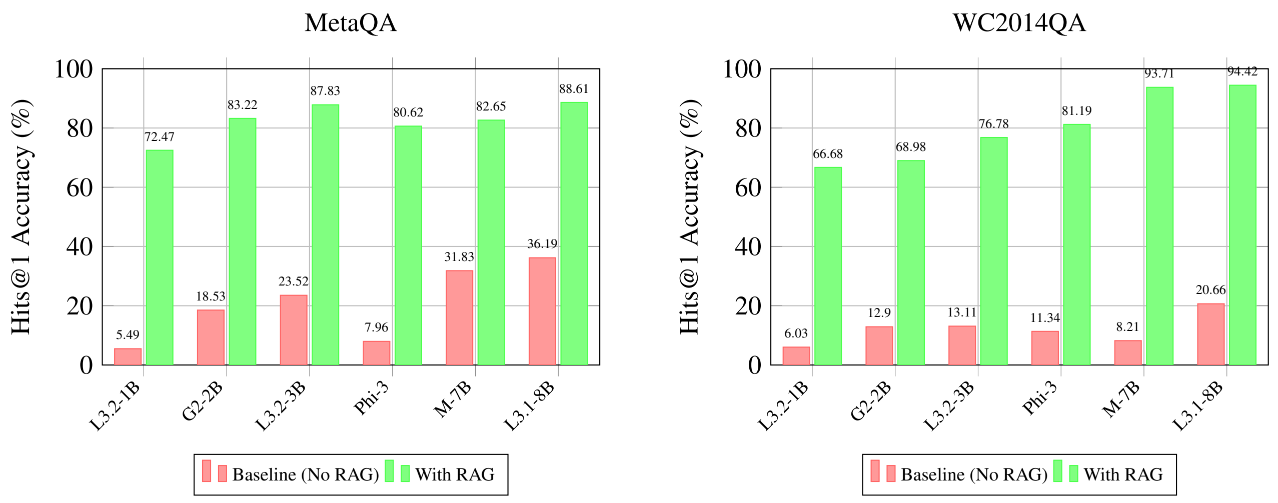

Figure 1: RAG accuracy improvement across both datasets. All models show substantial gains, with smaller models benefiting proportionally more. MetaQA shows 13× improvement for Llama 3.2 1B, WC2014QA shows 11.4× for Mistral-7B.

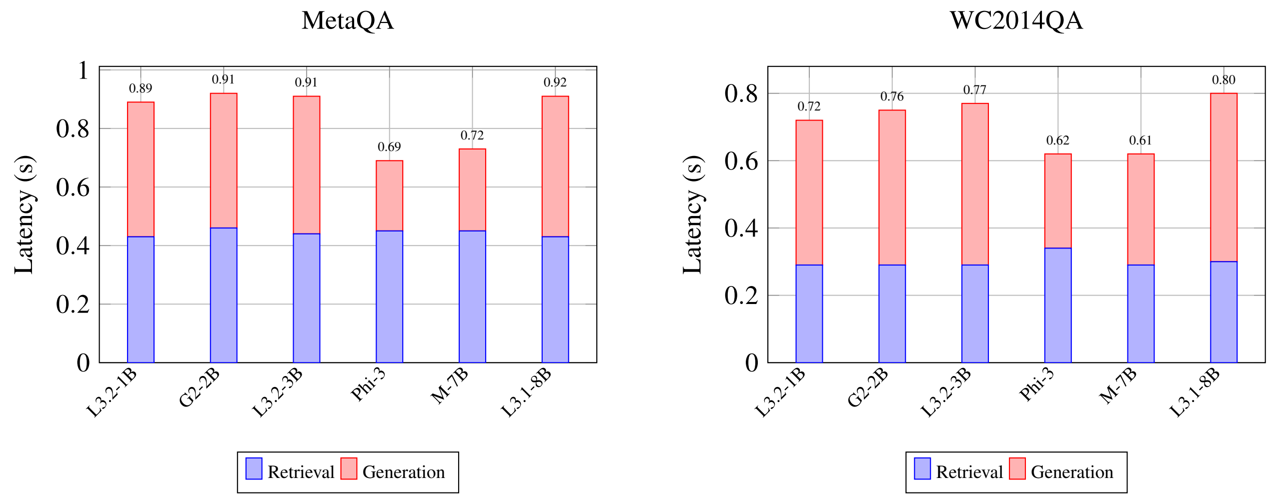

Figure 2: Latency breakdown per query. Retrieval latency is largely model-independent under the same service configuration; generation latency varies with model size and architecture. Phi-3 and Mistral-7B exhibit fastest generation.

pactful.

Limitations. Our evaluation is limited to two extractive QA benchmarks, which constrains generalization to other task types. All experiments use deterministic decoding and a fixed retrieval configuration; real-world deployments may differ. Absolute latency values are environment-dependent, and we do not perform formal statistical significance testing, although the large evaluation sets provide stable estimates. Overall, our results suggest that RAG can democratize the deployment of capable language models: with strong retrieval and careful model selection, compact models (2B–7B) can recover most attainable accuracy while remaining feasible on consumer-grade hardware.

Acknowledgments

We acknowledge computational resources provided by the Institute of Computer Science, Foundation for Research and Technology – Hellas (ICS-FORTH). We thank Iordanis Sapidis for making the SemanticRAG framework available for experimental evaluation.

References

[1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, *et al.*, "Retrieval-augmented generation for knowledge-intensive NLP tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[2] Y. Zhang, H. Dai, Z. Kozareva, A. J. Smola, and L. Song, "Variational reasoning for question answering with knowledge graph," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[3] A. Bordes, N. Usunier, S. Chopra, and J. Weston, "Large-scale simple question answering with memory networks," *arXiv preprint arXiv:1506.02075*, 2015.

[4] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," in *Proceedings of the 2020 Conference on Empirical*

Methods in Natural Language Processing (EMNLP), pp. 6769–6781, 2020.

[5] Z. Ma *et al.*, "Fine-tuned language models are continual learners," *arXiv preprint arXiv:2205.12393*, 2023.

[6] W. Xiong, X. L. Li, S. Iyer, J. Du, P. Lewis, W. Y. Wang, Y. Mehdad, W.-t. Yih, S. Riedel, D. Kiela, *et al.*, "Answering complex open-domain questions with multi-hop dense retrieval," in *International Conference on Learning Representations*, 2021.

[7] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, *et al.*, "Mistral 7B," *arXiv preprint arXiv:2310.06825*, 2023.

[8] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, *et al.*, "Gemma: Open models based on gemini research and technology," *arXiv preprint arXiv:2403.08295*, 2024.

[9] M. Abdin, S. A. Jacobs, A. A. Awan, J. Aneja, A. Awadallah, H. Awadalla, N. Bach, A. Bahree, A. Bakhtiari, H. Behl, *et al.*, "Phi-3 technical report: A highly capable language model locally on your phone," *arXiv preprint arXiv:2404.14219*, 2024.

[10] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, *et al.*, "The Llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[11] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, *et al.*, "Natural questions: a benchmark for question answering research," in *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 453–466, 2019.

[12] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer, "TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 1601–1611, 2017.

[13] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," in *NeurIPS Workshop on Energy Efficient Machine Learning and Cognitive Computing*, 2019.

[14] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "TinyBERT: Distilling BERT for natural language understanding," *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4163–4174, 2020.

[15] Y. Li, S. Bubeck, R. Eldan, A. Del Giorno, S. Gunasekar, and Y. T. Lee, "Textbooks are all you need II: phi-1.5 technical report," *arXiv preprint arXiv:2309.05463*, 2023.

[16] M. Bellagente, D. Phung, J. Tow, D. Nguyen, S. Mahan, A. Tseng, and S. Biderman, "StableLM 3B 4E1T: Technical report," *arXiv preprint arXiv:2402.17834*, 2024.

[17] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," in *International Conference on Learning Representations*, 2021.

[18] G. Izacard and E. Grave, "Leveraging passage retrieval with generative models for open domain question answering," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 874–880, 2021.

[19] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[20] I. Sapidis, V. Zervos, M. Mountantonakis, and Y. Tzitzikas, "Interactive and provenance-aware search and QA over documents using LLMs, RAG and knowledge graph verbalization," in *Proceedings of the 29th International Conference on Theory and Practice of Digital Libraries (TPDL 2025)*, (Tampere, Finland), September 2025.