Less LLM, More Documents: Searching for Improved RAG

Jingjie Ning(✉), Yibo Kong, Yunfan Long, and Jamie Callan

School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
{jening,yibok,justinlo}@andrew.cmu.edu, callan@cs.cmu.edu

Abstract. Retrieval-Augmented Generation (RAG) couples document retrieval with large language models (LLMs). While scaling generators often improves accuracy, it also increases inference and deployment overhead. We study an orthogonal axis: enlarging the retriever's corpus, and how it trades off with generator scale. Across multiple open-domain QA benchmarks, corpus scaling consistently strengthens RAG and can in many cases match the gains of moving to a larger model tier, though with diminishing returns at larger scales. Small- and mid-sized generators paired with larger corpora often rival much larger models with smaller corpora; mid-sized models tend to gain the most, while tiny and very large models benefit less. Our analysis suggests that these improvements arise primarily from increased coverage of answer-bearing passages, while utilization efficiency remains largely unchanged. Overall, our results characterize a corpus–generator trade-off in RAG and provide empirical guidance on how corpus scale and model capacity interact in this setting.

Keywords: Retrieval-Augmented Generation · Passage Retrieval · Large Language Models · Corpus Scaling · Resource-Constrained Inference

1 Introduction

Retrieval-Augmented Generation (RAG) [6,17] augments large language models (LLMs) with retrieved evidence and is widely used for open-domain question answering (QA) [8,28,30]. While much prior work improves RAG by scaling the generator, larger models typically incur higher inference and deployment overhead [7,19,21]. In parallel, the retriever determines what evidence is available to the generator and can help mitigate hallucinations [17,25], yet how retriever-side scaling interacts with generator scale is not well understood.

We study this interaction by varying the retrieval corpus scale while holding the retrieval pipeline and evidence budget fixed. Concretely, we combine randomly-shardable retrieval over ClueWeb22 [20] with open-source Qwen3 models [33] spanning 0.6B–14B parameters, and evaluate on NQ, TriviaQA,

J. Ning, Y. Kong and Y. Long—Equal Contributions.

R. Campos et al. (Eds.): ECIR 2026, LNCS 16483, pp. 598–613, 2026.
https://doi.org/10.1007/978-3-032-21289-4_38

and WebQ [1,13,16]. Our results show that enlarging the corpus consistently improves RAG and can, in this setting, partially offset smaller generators: for example, a 1.7B model with a 4× larger corpus outperforms a 4B model, and a 4B model with a 2× larger corpus consistently outperforms an 8B model.

Overall, our work characterizes a corpus–generator trade-off in RAG and provides empirical guidance on (i) when corpus scaling can match gains from scaling the generator, (ii) which model-size regimes benefit most from additional corpus, and (iii) why these gains arise, via analyses of answer-bearing evidence coverage and context utilization.

2 Related Work

A substantial body of research has focused on enhancing the intrinsic capabilities of LLMs. Instruction tuning [31,34] and prompt engineering [9,22] improve alignment with user queries, while scaling model size generally yields higher accuracy across diverse tasks [3,14]. However, ever-larger LLMs (e.g., PaLM with 540B parameters [4]) incur prohibitive computational costs, which limits their practicality in many settings.

In parallel, retrieval-augmented models have emerged as an alternative path to scaling. Retrieval-augmented language models (RALMs) such as RETRO [2] and Atlas [12] demonstrate that enlarging inference-time datastores consistently improves performance: relatively small generators paired with massive retrieval memory can outperform much larger LM-only baselines. Shao et al. [24] further confirmed this monotonic trend. Unlike modular RAG, which decouples retriever and generator, RALMs integrate retrieval through pretraining-time vector memories, requiring retriever-generator co-training. In contrast, we focus on modular RAG with a fixed evidence budget and study corpus–generator trade-offs across model-size regimes, rather than monotonic retrieval scaling in integrated RALMs.

Modular RAG instead relies on external corpora that can be scaled independently of the generator. This line of work has progressed from Dense Passage Retrieval (DPR) [15] to efficiency-oriented methods such as ANCE [32] and Contriever [11], which make retrieval over very large corpora feasible. These advances enabled a shift from early Wikipedia-only setups to broader and more diverse corpora such as LoTTE [23] and BEIR [27]. However, while the community has implicitly moved toward increasingly larger corpora, the direct impact of corpus growth itself has not yet been systematically and comprehensively examined.

More recently, studies have begun to explore broader factors influencing RAG, including model size, corpus scale, and context size [18,29]. However, these analyses remain fragmented and primarily descriptive, typically isolating single variables. Importantly, they do not provide a principled understanding of how corpus size and LLM size interact, leaving the corpus-generator trade-off essentially unexplored. In particular, prior studies have not jointly examined retrieval corpus expansion and LLM size, leaving open the fundamental question of how corpus-generator trade-offs shape overall system performance.

3 Methodology

We present a systematic framework to analyze corpus–generator trade-offs in RAG, asking when scaling the retrieval corpus can compensate for smaller LLMs and how this interaction varies across model sizes. Our design centers on two questions: (i) *corpus scaling as compensation*—whether enlarging the corpus enables smaller generators to match or surpass larger ones; and (ii) *differential effects across LLMs*—how benefits from corpus expansion change with model capacity.

3.1 Retriever: Corpus Scaling

Let $\mathcal{C}$ denote a fixed corpus. We simulate corpus scaling by randomly partitioning $\mathcal{C}$ into N disjoint shards $\{S_1, \ldots, S_N\}$ of approximately equal size:

$$\Pi(\mathcal{C}) \;\rightarrow\; \{S_1, S_2, \ldots, S_N\}, \quad S_i \cap S_j = \varnothing \;\; \forall\, i \neq j, \quad \bigcup_{i=1}^{N} S_i = \mathcal{C}.$$

A corpus scale $n \in \{1, \ldots, N\}$ is realized by activating n shards; we use the canonical prefix $\mathcal{C}^{(n)} = \bigcup_{i=1}^{n} S_i$, so that $|\mathcal{C}^{(n)}| \propto n$. For a query q, the retriever operates on $\mathcal{C}^{(n)}$ to retrieve top-k documents, segment them into chunks, rerank, and pass the top-m chunks to the generator.

3.2 Generator

We consider a family of generators $\{M_x\}$, where M_x denotes a model with parameter size x drawn from the same architecture family. Each generator takes as input a fixed template consisting of the query and retrieved chunks.

3.3 Trade-Off Formalization

We adopt a full-factorial design pairing each corpus scale $n \in \{1, \ldots, N\}$ with each generator M_x. Throughout, we keep retrieval, prompting, and decoding settings fixed, so that only corpus scale n and model size x vary. Let $P_m(n, x)$ denote the evaluation score under metric $m \in \{\mathrm{F1}, \mathrm{EM}\}$.

To quantify *corpus-as-compensation*, we define

$$n^{\star}(x_{small} \rightarrow x_{large}) \;:=\; \min_{m \in \{\mathrm{F1},\, \mathrm{EM}\}} \; \min \left\{ n \;:\; P_m(n, x_{small}) \;\geq\; P_m(1, x_{large}) \right\},$$

the smallest corpus scale where a smaller generator $M_{x_{small}}$ matches the 1-shard baseline ($n = 1$) of a larger generator $M_{x_{large}}$. We report $n^{\star}$ and efficiency curves across (n, x) to characterize corpus–generator trade-offs. Section 4 details datasets, metrics, and constants.

4 Experiment

Building on the methodology outlined in Sect. 3, we conducted a series of controlled experiments across different corpus scales to systematically evaluate our research questions.

4.1 Benchmarks

We evaluate on three open-domain QA benchmarks: NQ [16] (1,769 real Google queries from *open-domain test set*), TriviaQA [13] (1,000 encyclopedic questions randomly sampled from the 9.51k *rc.web test split*), and WebQ [1] (2,032 Google Suggest queries with Freebase annotations from *standard test set*).

Scope and Limitation. Our evaluation targets open-domain QA with a large web corpus (ClueWeb22-A), which may overlap with modern LLM pretraining data; thus our results should be interpreted within this open-domain setting. Although our later analyses condition on questions that are initially incorrect without retrieval to isolate retrieval-driven gains, this setting still differs from domain-gap or enterprise RAG where the corpus is largely unseen during pretraining; thus, extrapolations should be made with caution.

4.2 Evaluation Metrics

We report Exact Match (EM) and token-level F1 using the official evaluation scripts. EM is the percentage of predictions that exactly match any normalized gold answer string. F1 is the average token-overlap F1 between the prediction and the gold answers, taking the maximum over gold answers for each question. Our analysis primarily focuses on these two metrics throughout the paper.

4.3 Retriever: Implementation Details

Corpus and Sharding. We use a 30% subset of ClueWeb22-A [20], comprising ∼264M English documents. The corpus is partitioned into 12 balanced shards of ∼22M documents each, via randomized local assignments to reduce topical skew (though some popularity bias may persist).

Retrieval Pipeline. We use `MiniCPM-Embedding-Light` [10] for dense passage encoding and build ANN indices with `DiskANN` [26], which supports fast multi-shard retrieval and has been used in other large-scale ClueWeb22-A retrievers [5]. As an auxiliary check, we also encoded the corpus with a similarly sized `Qwen3-Embedding-0.6B` encoder [35] and observed consistent corpus-scaling trends. For each corpus scale n, the retriever searches the active shards to retrieve the top-10 documents, segments them into overlapping chunks, and reranks the candidates; we then pass the top-8 chunks to the generator. For reranking, we use the higher-capacity `MiniCPM-Embedding` model from the same embedding family to improve ranking quality, while keeping all retrieval and reranking settings fixed across n to isolate corpus scaling effects.

4.4 Generator: Implementation Details

We instantiate $\{M_x\}$ using the open-source Qwen3 base series [33]: `Qwen3-0.6B`, `1.7B`, `4B`, `8B`, and `14B`. These models share the same tokenizer and a consistent architecture family as described in the Qwen3 technical report, spanning over an order of magnitude in parameter scale.

All models use identical prompting templates and decoding settings, enabling controlled comparisons of corpus–model trade-offs. While Qwen3 sizes share the same tokenizer and architecture family, different scales may involve minor training-recipe differences; thus, our results should be interpreted as empirical trends across sizes within the same Qwen3 series. We observed consistent trends in preliminary experiments with Qwen2.5 and early LLaMA models, but omit them due to the lack of a similarly homogeneous series and slower inference.

5 Results and Analysis

5.1 The Effect of Corpus Scaling as Compensation

We examine whether enlarging the retriever corpus can compensate for smaller LLMs, enabling them to match or surpass larger model. We parameterize corpus scale by the number of active shards n, where each shard indexes ∼22M documents from ClueWeb22-A; scaling the corpus corresponds to increasing n.

Table 1. Natural Questions. Shaded cells mark the first scale where a smaller model catches up to the next model's $n = 1$ baseline, i.e., $n^\star(x_{small} \rightarrow x_{large})$.

Corpus	$M_{0.6B}$		$M_{1.7B}$		M_{4B}		M_{8B}		M_{14B}	
$\times n$	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
1	25.37	16.68	33.44	23.74	40.27	28.49	42.42	30.24	44.26	32.50
×2	29.90	19.50	38.85	28.32	**44.16**	**32.33**	**45.59**	**33.35**	46.96	34.65
×3	30.99	20.75	**41.00**	**29.96**	46.00	34.31	46.88	34.71	48.94	36.69
×4	32.72	21.54	41.08	29.85	46.33	34.43	48.16	35.56	48.98	37.08
×5	33.26	22.84	41.49	30.19	46.49	34.37	48.40	35.39	49.51	37.14
×6	**34.00**	23.69	42.15	30.36	46.92	34.09	48.89	36.24	49.57	37.03
×7	34.62	**23.74**	42.15	30.92	47.25	34.77	48.04	35.44	49.54	36.80
×8	34.72	24.19	42.12	31.09	46.91	34.60	48.04	35.73	49.34	36.80
×9	34.49	23.97	42.04	30.75	46.64	34.31	48.27	35.61	49.36	36.63
×10	35.35	24.59	42.86	31.32	47.08	34.43	48.62	36.07	49.75	37.37
×11	35.48	25.10	42.78	30.98	47.43	34.71	49.06	36.01	50.38	37.59
×12	35.56	25.10	43.14	31.32	47.89	34.88	48.73	35.78	50.43	37.70

We evaluate five Qwen3 variants ($M_{0.6B}$, $M_{1.7B}$, M_{4B}, M_{8B}, M_{14B}) and, for each model, vary n by cumulatively activating shards under the same retrieval and prompting protocol, enabling controlled cross-model comparisons.

Compensation Effect. Our results show clear evidence that corpus expansion enables smaller models to match or even outperform larger counterparts. On NQ, as shown in Table 1, we find that the smallest model needs more corpus to surpass the larger model $n^\star(0.6\text{B} \rightarrow 1.7\text{B}) = 6$. For larger models, it is much easier: $n^\star(4\text{B} \rightarrow 8\text{B}) = 2$ and $n^\star(8\text{B} \rightarrow 14\text{B}) = 2$. These results indicate that, under our setup, scaling corpus size can be a more effective and efficient lever than simply scaling LLM size. Figures 1 and 2 visualize these catch-up points.

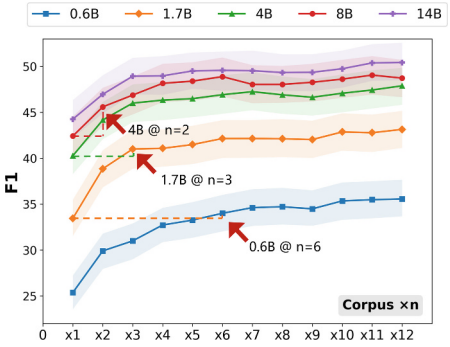


Fig. 1. F1 Gains from Scaling on NQ.

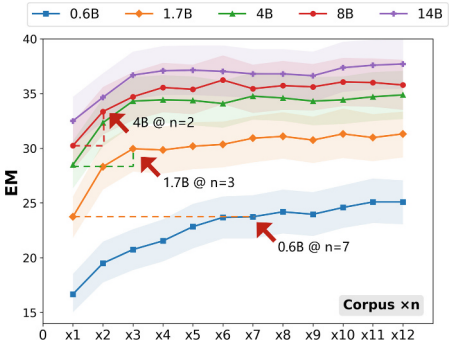


Fig. 2. EM Gains from Scaling on NQ.

The same trend holds on TriviaQA and WebQ (Table 2). In the tiny-model regime (sub-2B parameters), corpus scaling is inefficient: e.g., $n^\star(0.6\text{B} \rightarrow 1.7\text{B}) = 10$ and $n^\star(1.7\text{B} \rightarrow 4\text{B}) = 7$ on TriviaQA, indicating weaker scaling efficiency at small model sizes. In contrast, once the generator reaches medium to large scale (roughly 4∼14B parameters), **doubling** the corpus is typically sufficient to catch up with the next-tier model. For example, $n^\star(4\text{B} \rightarrow 8\text{B}) = 2$ and $n^\star(8\text{B} \rightarrow 14\text{B}) = 2$ on NQ and TriviaQA, and at most $n^\star = 3$ on WebQ. Detailed results for TriviaQA and WebQ are provided in Table 3 and Table 4 in the Appendix; using `Qwen3-Embedding` as the retriever on NQ yields nearly identical $n^\star$ patterns, suggesting this trend transfers across embedding models.

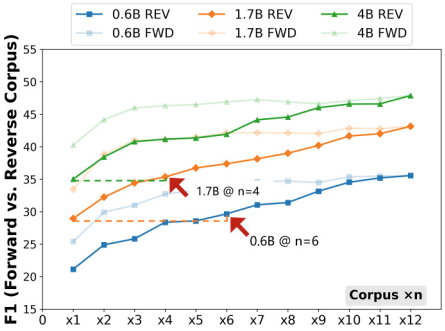
Table 2. $n^\star$ across datasets

$n^\star$	NQ	TriviaQA	WebQ	NQ Qwen3-Emb
0.6B→1.7B	6	10	9	5
1.7B→4B	3	7	4	3
4B→8B	2	2	3	2
8B→14B	2	2	1	2

Serving-Time Compute Proxy. We provide a simple *serving-time* compute proxy by estimating FLOPs per query for retrieval and generation. Under a representative setting (top-$K = 10$, $L = 5K = 50$, dot-product retrieval with $D = 1024$; generation with $S \approx 2048$ prompt tokens and $T \approx 64$ output tokens), a single-shard DiskANN retrieval costs ∼7.168 MFLOPs/query, while Qwen3-0.6B and Qwen3-1.7B generation cost ∼1.736 TFLOPs/query and ∼5.953 TFLOPs/query,

respectively. This orders-of-magnitude gap is already sufficient to show that, in our setup, arithmetic compute is dominated by generation rather than retrieval. We focus on *inference-time* compute because our experiments vary serving-side design choices such as corpus scaling and generator size. This proxy does not account for one-time costs such as LLM pretraining or index/shard construction.

Corpus Quality vs. Quantity. Shards are balanced in size but not perfectly uniform, since randomization was applied locally rather than globally. Consequently, shard order may retain residual crawl/popularity gradients (e.g., earlier shards being slightly more likely to contain higher-traffic or higher-quality pages). To probe sensitivity to corpus quality, we reverse the shard order at retrieval time, replacing $S_1, \ldots, S_n$ with $S_{N-n+1}, \ldots, S_N$, which delays the "head" content. As expected, this yields a modest drop in absolute end-task QA performance (Fig. 3, left).



$n^\star$	NQ-FWD	**NQ-REV**
0.6B→1.7B	6	6
1.7B→4B	3	4
4B→8B	2	2
8B→14B	2	2

Fig. 3. F1 and Catch-up Thresholds under **Reversed** Corpus Scaling. Left: F1 when using forward (FWD) vs. reversed (REV) corpus scaling order. Right: corresponding catch-up thresholds.

In other words, lower average corpus quality shifts performance downward, yet the *relative* additional corpus needed for a smaller model to catch up with the next larger model remains largely stable. This supports our conclusion: corpus quality affects absolute accuracy, but scaling corpus quantity can still enable smaller generators to match or overtake larger ones with similar additional context. For consistency, we report all subsequent results using the **forward** order.

Why Does Corpus Scaling Improve RAG? At the micro level, corpus scaling increases the likelihood that retrieved passages explicitly contain the gold answer. With a small corpus scale ($n = 1$), retrieved chunks often lack factual mentions. The larger n brings the direct answer terms into the context, providing the generator with grounding evidence as intended in RAG.

Case Analysis

Question: "Obey your thirst" is the advertising slogan for which soft drink? **Answer:** Sprite

Retrieved Fragment with Corpus scale $n = 4$

Retrieved passage:
...Obey Your Thirst (Oct 1, 1997). Ever heard that catchy slogan for Sprite? "Image is nothing. Thirst is everything. Obey your thirst." In the summer of 1996, Coca-Cola, who manufactures Sprite products, was looking to change the image of its sparkling soda..."

At the aggregate level, we measure the probability that at least one of the top-8 retrieved chunks fed into the generator contains a gold answer string, using the same normalization/aliasing as our EM metric. We refer to this probability as the **Gold Answer Coverage Rate**, which upper-bounds achievable EM under perfect reasoning. Figure 4 shows two key findings:

- **Monotonic Growth.** Gold answer coverage often rises consistently with corpus scale, confirming that corpus expansion increases the likelihood of providing usable evidence.
- **Dataset Variation.** The magnitude of this benefit differs across benchmarks. TriviaQA exhibits substantially higher coverage than NQ or WebQ, indicating stronger overlap between its information needs and ClueWeb22.

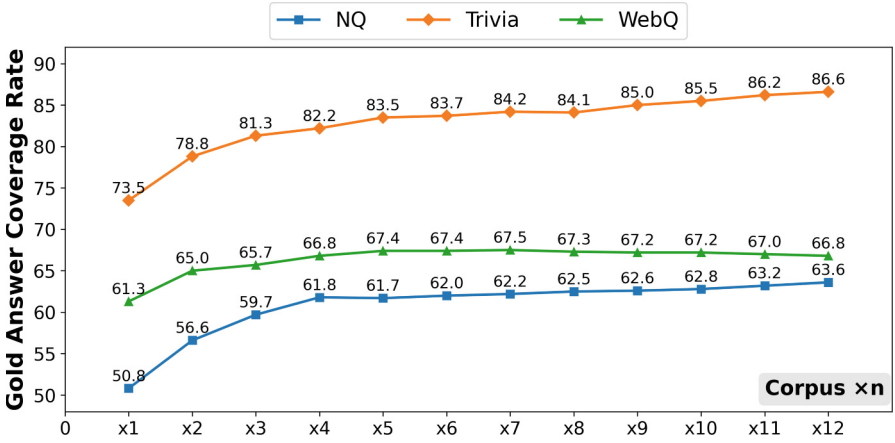


Fig. 4. Gold Answer Coverage Rate for Forward Scaling.

5.2 Differential Effects Across LLM Sizes

To analyze how retrieval corpus size differentially affects LLMs of varying scales, we focus on questions that are *initially unanswerable* without retrieval and examine how performance changes as corpus size increases. Since correctness is most relevant here, we primarily rely on the Exact Match (EM) metric.

Classification Methodology. Let $n \in \{0, 1, \ldots, 12\}$ denote the corpus size in shards, where $n = 0$ represents the no-retrieval baseline. We define the **Context-Benefited Success Rate (CB) at shard** n as

$$\mathrm{CB@}n := \Pr\big(EM_{n\text{-shard}} = 1 \,\big|\, EM_{0\text{-shard}} = 0\big)$$

i.e., the *empirical proportion* of initially unanswerable questions that become answerable once n shards are available. By construction, $\mathrm{CB@}0 = 0$. For $n \geq 1$, we also report the marginal improvement.

$$\Delta_n \;:=\; \mathrm{CB@}n \;-\; \mathrm{CB@}(n-1)$$

which captures the fraction of initially unanswerable questions solved at shard n. *Note* that although CB@n is computed cumulatively, it is an empirical statistic and need not be strictly monotonic in n (e.g., additional shards may introduce noise).

Although CB@n reflects the gains realized, it is bounded above by *Gold Answer Coverage Rate* at shard n. We define the **Utilization Ratio** as

$$\mathrm{Ratio@}n \;:=\; \frac{\mathrm{CB@}n}{\mathrm{Coverage@}n}$$

This ratio quantifies an LLM's ability to take advantage of the retrieved evidence: Coverage@n indicates how often the gold answer is retrievable, while CB@n records how often the model succeeds when given the opportunity.

CB excludes questions already correct at $n = 0$ (Known), so it isolates retrieval-driven gains by removing parametric knowledge effects. The **Known Rate** in Fig. 5 summarizes how much each model can answer without retrieval.

Figure 6 reveals a consistent and model-invariant pattern for how the scaling helps answer questions that are *initially unanswerable* without context, with similar trends observed on TriviaQA and WebQ (see the Appendix).

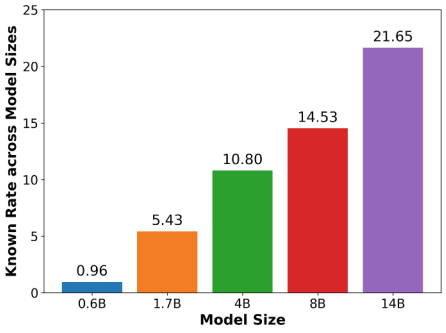


Fig. 5. Known Rate on NQ.

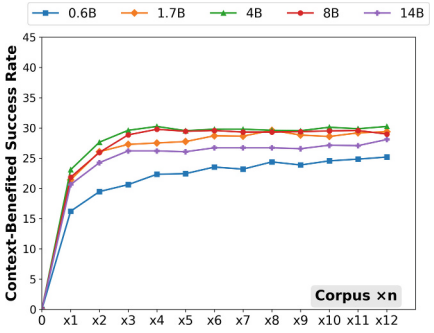


Fig. 6. CB Rate on NQ (FWD).

Initial Jump, Subsequent Growth, and Saturation

The Critical Impact of Initial Retrieval. The most dramatic performance jump occurs when moving from zero context to a single shard across models, with Δ_1 ranging from 16.2% to 20.6%, whereas Δ_2 ranges from only 2.8% to 4.4% (Fig. 6). This dominance of initial retrieval persists even with low-quality corpora: for $M_{1.7B}$ with reversed shard ordering, $\Delta_1 = 16.6\%$ versus $\Delta_2 = 2.6\%$. This highlights the primary benefit of RAG: even a small corpus immediately fills a substantial fraction of knowledge gaps.

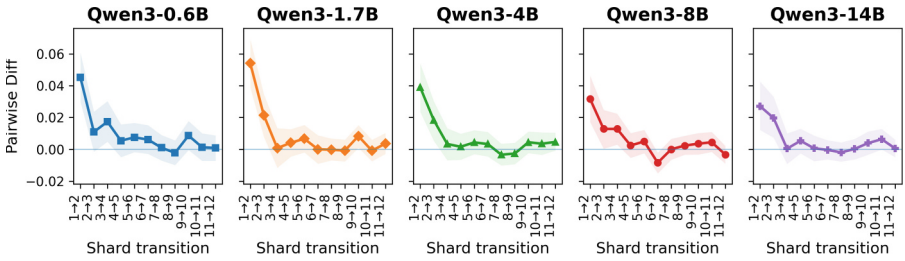


Fig. 7. F1 Pairwise Bootstrap on NQ.

Model-invariant growth and saturation. Across all LLM sizes, adjacent-shard improvements in downstream QA (F1) exhibit a highly consistent pattern. The paired bootstrap differences from shard n to $n+1$ show large, reliably positive gains in the early steps, followed by rapid decay toward a near-zero regime (Fig. 7). For all model sizes, the 95% CI typically begins to include zero around the 3→4 or 4→5 transition, indicating diminishing returns and that later fluctuations are often within statistical noise rather than systematic improvement; we observe the same conclusion when computing paired differences using EM.

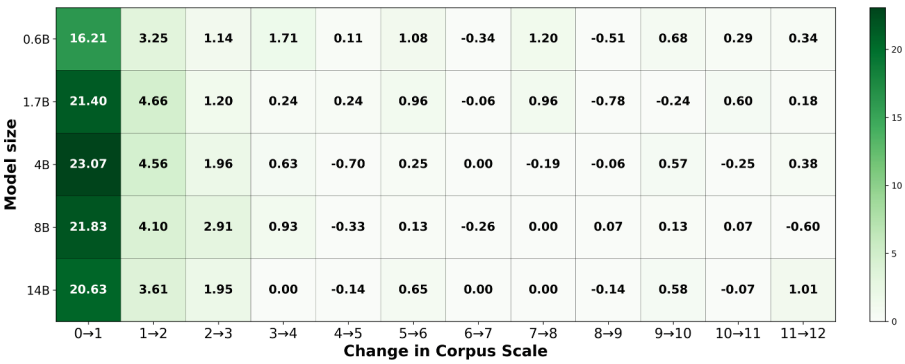


Fig. 8. Per-shard CB gains Δ_n on NQ (FWD).

A consistent picture emerges for retrieval-driven gains on initially unanswerable questions. Corpus scaling yields a qualitatively similar CB trajectory across models: a sharp first jump, sustained gains up to roughly $n \approx 6$, and diminishing returns thereafter. The per-shard increments Δ_n also follow a nearly identical pattern across models: peaking early and tapering to near zero (Fig. 8). Together, these results suggest a size-invariant retrieval effect: additional shards do not yield systematically larger marginal gains for larger generators than for smaller ones. In practice, corpus expansion mainly shifts performance upward without materially changing the shape of the growth curve.

LLM Context Utilization Remains Stable Across Corpus Scales. As shown in Fig. 9, the Utilization Ratio stays approximately constant across corpus scales and varies only slightly across models. Although both CB@n and Coverage@n increase with n, their ratio remains stable, indicating that corpus scaling primarily improves answer-bearing evidence coverage rather than changing how efficiently generators use the provided context. As a result, gains largely track the increased availability of relevant evidence as the corpus grows.

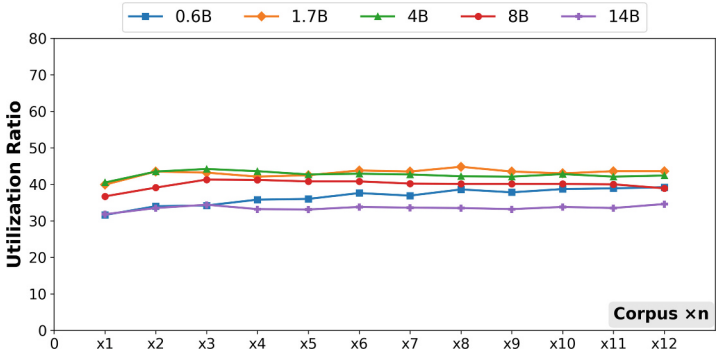


Fig. 9. Utilization Ratio across models on NQ (FWD).

Non-monotonic Context Utilization. One might expect larger LLMs to always leverage retrieved context more effectively, but Fig. 9 shows otherwise. Mid-sized models ($M_{1.7B}$ and M_{4B}) achieve the highest Utilization Ratio (peaking near 42%), while the largest M_{14B} lags behind. This suggests that context utilization does not grow monotonically with model size, and mid-sized models can sometimes exploit retrieval more efficiently than their larger counterparts.

6 Conclusion

In this paper, we asked whether scaling the retrieval corpus can substitute for scaling the generator in RAG, and how corpus size interacts with model size

under a fixed evidence budget. Using controlled evaluations on NQ, TriviaQA, and WebQ with standardized prompting and context formatting, we characterize a corpus–generator trade-off while holding the presented evidence constant.

Our results indicate that, in this open-domain setting, corpus scaling can often *partially offset* model downsizing. Across datasets, enlarging the corpus is a reliable lever: smaller or mid-sized generators paired with larger corpora frequently approach, and sometimes exceed, the accuracy of larger models under the same evidence budget. In several settings, moving up corpus tiers closes the gap of one to two model-size tiers, suggesting that increasing corpus scale can in many cases yield gains comparable to moving to a larger generator tier.

A consistent mechanism explains these gains: *improvements are primarily driven by increased coverage of answer-bearing evidence rather than changes in utilization efficiency.* As the corpus grows, the likelihood that retrieved passages contain the gold answer increases consistently, whereas the model's context-utilization ratio remains roughly stable across shard counts and model sizes; we observe the same qualitative corpus-scaling trends under an auxiliary retriever built with a different encoder, suggesting the main findings are not specific to a single embedding model. Thus, scaling mainly works by raising the hit rate of relevant evidence, instead of altering how effectively models exploit the provided context. At the same time, both the paired bootstrap differences between adjacent shard counts and the per-shard CB increments Δ_n show that marginal gains rapidly decay and become indistinguishable from zero beyond roughly a $5\times$ corpus increase, indicating clear diminishing returns at larger scales.

From a practical perspective, our findings suggest that retrieval-side scaling is an important design axis when additional corpus is available and retrieval costs are acceptable under deployment constraints. Under our evidence budget, a simple serving-time FLOPs proxy indicates an orders-of-magnitude compute gap between single-shard retrieval and generation, suggesting that overall arithmetic compute is typically dominated by the generator (notwithstanding retrieval's memory/IO and indexing overheads). In our experiments, mid-sized generators tend to convert improved evidence coverage into end-task gains more effectively than very small models (which require steep corpus expansions) and very large models (which yield smaller marginal gains). While we mainly focus on the Qwen3 family due to the lack of other open-source LLM series with homogeneous, wide-ranging variants, we hope to extend this analysis to additional model and retriever families, as well as more domain-specific and siloed corpora, in future work. Finally, tracking *gold-answer coverage* together with the *Utilization Ratio* provides practical diagnostics for when increasing retrieval scale is likely to translate into downstream improvements; when corpus expansion is limited (e.g., enterprise or specialized domains), the same evidence-centric view suggests a *coverage-first* strategy via domain-focused curation and retrieval quality improvements, rather than adding low-yield documents.

Disclosure of Interests. The authors have no competing interests to declare.

Appendix

See Figs. 10 and 11.

Table 3. $n^{\star}(x_{small} \rightarrow x_{large})$ for TriviaQA.

Corpus	$M_{0.6B}$		$M_{1.7B}$		M_{4B}		M_{8B}		M_{14B}	
$\times n$	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
1	44.74	37.20	57.40	49.80	66.46	59.80	69.86	62.90	73.16	66.60
$\times 2$	49.73	41.00	62.68	55.10	**73.17**	**65.50**	**76.32**	**68.80**	77.72	71.10
$\times 3$	51.86	43.60	64.90	57.20	75.99	68.50	77.59	69.90	79.59	72.50
$\times 4$	53.20	45.10	66.18	59.00	76.56	69.60	78.04	71.20	80.75	74.00
$\times 5$	54.79	46.40	66.08	59.30	76.86	69.40	78.00	71.00	80.68	73.90
$\times 6$	55.26	47.55	66.01	58.16	76.60	69.77	77.58	70.57	80.70	74.07
$\times 7$	55.59	47.30	**66.89**	59.30	76.68	69.80	78.11	70.90	80.55	73.90
$\times 8$	56.52	48.00	67.31	59.50	77.73	69.40	78.27	71.30	80.77	74.20
$\times 9$	55.62	47.20	68.32	**60.90**	77.61	70.40	79.30	72.50	81.10	74.40
$\times 10$	**57.61**	48.90	68.23	60.90	77.92	70.90	79.88	73.30	81.56	74.90
$\times 11$	58.74	**50.40**	68.60	61.40	78.13	71.00	79.68	72.90	82.13	75.50
$\times 12$	57.86	49.05	68.44	61.16	77.89	70.77	80.05	73.27	82.00	75.28

Table 4. $n^{\star}(x_{small} \rightarrow x_{large})$ for WebQuestions.

Corpus	$M_{0.6B}$		$M_{1.7B}$		M_{4B}		M_{8B}		M_{14B}	
$\times n$	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
1	27.63	14.81	33.91	18.01	37.01	20.28	38.32	**21.70**	38.68	21.65
$\times 2$	28.95	15.70	35.40	19.09	38.20	20.92	**38.92**	21.99	39.46	22.19
$\times 3$	29.99	16.09	35.41	19.64	**38.81**	20.96	39.44	22.60	40.49	22.93
$\times 4$	31.37	17.62	35.96	**20.28**	39.47	21.60	40.17	23.52	41.32	24.02
$\times 5$	31.68	17.66	36.35	20.28	39.59	21.55	40.29	23.38	41.08	23.77
$\times 6$	31.58	17.32	36.47	20.08	39.98	21.66	40.24	23.13	41.22	23.67
$\times 7$	31.37	17.32	36.46	20.03	39.65	21.65	40.12	22.63	41.25	23.52
$\times 8$	31.80	17.62	36.64	19.92	39.62	21.57	40.00	22.93	40.84	23.08
$\times 9$	32.33	**18.09**	36.62	20.34	39.36	21.45	39.79	22.83	40.57	22.74
$\times 10$	32.18	18.09	36.83	20.77	38.99	21.62	39.95	23.50	40.79	22.93
$\times 11$	31.98	17.91	36.61	20.57	39.20	21.51	39.94	22.58	40.58	22.88
$\times 12$	32.00	17.82	36.94	21.01	39.41	21.46	40.12	22.93	40.69	23.18

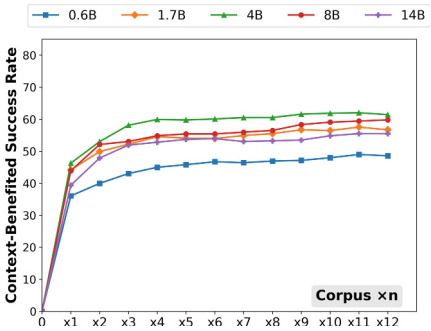


Fig. 10. CB Rate for **TriviaQA**.

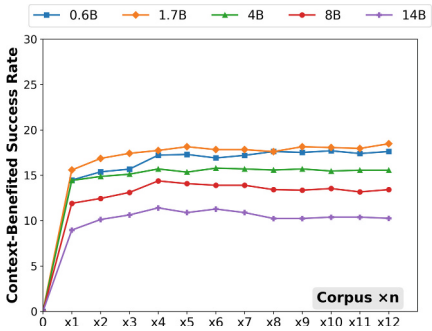


Fig. 11. CB Rate for **WebQ**.

References

1. Berant, J., Chou, A., Frostig, R., Liang, P.: Semantic parsing on Freebase from question-answer pairs. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1533–1544. Association for Computational Linguistics, Seattle, Washington, USA (2013). https://www.aclweb.org/anthology/D13-1160
2. Borgeaud, S., et al.: Improving language models by retrieving from trillions of tokens (2022). https://arxiv.org/abs/2112.04426
3. Brown, T.B., et al.: Language models are few-shot learners (2020). https://arxiv.org/abs/2005.14165
4. Chowdhery, A., et al.: Palm: scaling language modeling with pathways. J. Mach. Learn. Res. **24**(1) (2023)
5. Coelho, J., et al.: Deepresearchgym: a free, transparent, and reproducible evaluation sandbox for deep research (2025). https://arxiv.org/abs/2505.19253
6. Gao, Y., et al.: Retrieval-augmented generation for large language models: a survey (2024). https://arxiv.org/abs/2312.10997
7. Ghorbani, B., et al.: Scaling laws for neural machine translation. In: International Conference on Learning Representations (2022). https://openreview.net/forum?id=hR_SMu8cxCV
8. Gupta, S., Ranjan, R., Singh, S.N.: A comprehensive survey of retrieval-augmented generation (rag): evolution, current landscape and future directions (2024). https://arxiv.org/abs/2410.12837
9. He, Z., Jiang, H., Wang, Z., Yang, Y., Qiu, L.K., Qiu, L.: Position engineering: boosting large language models through positional information manipulation. In: Al-Onaizan, Y., Bansal, M., Chen, Y.N. (eds.) Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pp. 7333–7345. Association for Computational Linguistics, Miami, Florida, USA (2024). https://doi.org/10.18653/v1/2024.emnlp-main.417. https://aclanthology.org/2024.emnlp-main.417/
10. Hu, S., et al.: MiniCPM: unveiling the potential of small language models with scalable training strategies. In: First Conference on Language Modeling (2024). https://openreview.net/forum?id=3X2L2TFr0f
11. Izacard, G., et al.: Unsupervised dense information retrieval with contrastive learning (2021). https://doi.org/10.48550/ARXIV.2112.09118. https://arxiv.org/abs/2112.09118

12. Izacard, G., et al.: Atlas: few-shot learning with retrieval augmented language models. J. Mach. Learn. Res. **24**(1) (2023)
13. Joshi, M., Choi, E., Weld, D., Zettlemoyer, L.: TriviaQA: a large scale distantly supervised challenge dataset for reading comprehension. In: Barzilay, R., Kan, M.Y. (eds.) Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1601–1611. Association for Computational Linguistics, Vancouver, Canada (2017). https://doi.org/10.18653/v1/P17-1147. https://aclanthology.org/P17-1147/
14. Kaplan, J., et al.: Scaling laws for neural language models (2020). https://arxiv.org/abs/2001.08361
15. Karpukhin, V., et al.: Dense passage retrieval for open-domain question answering. In: Webber, B., Cohn, T., He, Y., Liu, Y. (eds.) Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 6769–6781. Association for Computational Linguistics, Online (2020). https://doi.org/10.18653/v1/2020.emnlp-main.550. https://aclanthology.org/2020.emnlp-main.550/
16. Kwiatkowski, T., et al.: Natural questions: a benchmark for question answering research. Trans. Assoc. Comput. Linguist. **7**, 452–466 (2019). https://doi.org/10.1162/tacl_a_00276. https://aclanthology.org/Q19-1026/
17. Lewis, P., et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS 2020. Curran Associates Inc., Red Hook, NY, USA (2020)
18. Li, S., Stenzel, L., Eickhoff, C., Bahrainian, S.A.: Enhancing retrieval-augmented generation: a study of best practices. In: Rambow, O., Wanner, L., Apidianaki, M., Al-Khalifa, H., Eugenio, B.D., Schockaert, S. (eds.) Proceedings of the 31st International Conference on Computational Linguistics, pp. 6705–6717. Association for Computational Linguistics, Abu Dhabi, UAE (2025). https://aclanthology.org/2025.coling-main.449/
19. Narayanan, D., et al.: Efficient large-scale language model training on GPU clusters using megatron-lm. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. SC 2021. Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3458817.3476209
20. Overwijk, A., Xiong, C., Liu, X., VandenBerg, C., Callan, J.: Clueweb22: 10 billion web documents with visual and semantic information (2022). https://arxiv.org/abs/2211.15848
21. Patterson, D., et al.: Carbon emissions and large neural network training (2021). https://arxiv.org/abs/2104.10350
22. Reynolds, L., McDonell, K.: Prompt programming for large language models: beyond the few-shot paradigm. In: Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems. CHI EA 2021. Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3411763.3451760
23. Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., Zaharia, M.: ColBERTv2: effective and efficient retrieval via lightweight late interaction. In: Carpuat, M., de Marneffe, M.C., Meza Ruiz, I.V. (eds.) Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 3715–3734. Association for Computational Linguistics, Seattle, United States (2022). https://doi.org/10.18653/v1/2022.naacl-main.272. https://aclanthology.org/2022.naacl-main.272/

24. Shao, R., et al.: Scaling retrieval-based language models with a trillion-token datastore. In: The Thirty-Eighth Annual Conference on Neural Information Processing Systems (2024). https://openreview.net/forum?id=iAkhPz7Qt3
25. Shi, W., et al.: REPLUG: retrieval-augmented black-box language models. In: Duh, K., Gomez, H., Bethard, S. (eds.) Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pp. 8371–8384. Association for Computational Linguistics, Mexico City, Mexico (2024). https://doi.org/10.18653/v1/2024.naacl-long.463. https://aclanthology.org/2024.naacl-long.463/
26. Subramanya, S.J., Devvrit, Kadekodi, R., Krishaswamy, R., Simhadri, H.V.: DiskANN: fast accurate billion-point nearest neighbor search on a single node. Curran Associates Inc., Red Hook, NY, USA (2019)
27. Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., Gurevych, I.: 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 (Round 2) (2021). https://openreview.net/forum?id=wCu6T5xFjeJ
28. Upadhyay, P., Agarwal, R., Dhiman, S., Sarkar, A., Chaturvedi, S.: A comprehensive survey on answer generation methods using NLP. Nat. Lang. Process. J. **8**, 100088 (2024). https://doi.org/10.1016/j.nlp.2024.100088
29. Vladika, J., Matthes, F.: On the influence of context size and model choice in retrieval-augmented generation systems. In: Chiruzzo, L., Ritter, A., Wang, L. (eds.) Findings of the Association for Computational Linguistics: NAACL 2025, pp. 6724–6736. Association for Computational Linguistics, Albuquerque, New Mexico (2025). https://doi.org/10.18653/v1/2025.findings-naacl.375. https://aclanthology.org/2025.findings-naacl.375/
30. Voorhees, E.M., Tice, D.M.: The TREC-8 question answering track. In: Gavrilidou, M., Carayannis, G., Markantonatou, S., Piperidis, S., Stainhauer, G. (eds.) Proceedings of the Second International Conference on Language Resources and Evaluation (LREC 2000). European Language Resources Association (ELRA), Athens, Greece (2000). https://aclanthology.org/L00-1018/
31. Wang, Y., et al.: Self-instruct: aligning language models with self-generated instructions. In: Rogers, A., Boyd-Graber, J., Okazaki, N. (eds.) Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 13484–13508. Association for Computational Linguistics, Toronto, Canada (2023). https://doi.org/10.18653/v1/2023.acl-long.754. https://aclanthology.org/2023.acl-long.754/
32. Xiong, L., et al.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=zeFrfgyZln
33. Yang, A., et al.: Qwen3 technical report (2025). https://arxiv.org/abs/2505.09388
34. Zhang, S., et al.: Instruction tuning for large language models: a survey. arXiv preprint arXiv:2308.10792 (2023)
35. Zhang, Y., et al.: Qwen3 embedding: advancing text embedding and reranking through foundation models. arXiv preprint arXiv:2506.05176 (2025)