JOURNAL OF BASIC SCIENCE AND ENGINEERING

Self-Correcting RAG Using LLM-Based Retrieval Verification

Vijay Dhaku Dhangar[1]

[1]School of Computer Science and Engineering, Sandip University, Nashik

DOI: https://doi.org/10.10399/JBSE.2026755184

Abstract

Retrieval-Augmented Generation (RAG) reduces hallucination by grounding large language model outputs in externally retrieved evidence, but the standard single-pass pipeline has no mechanism to detect or correct a failed retrieval: when retrieved passages are irrelevant, insufficient, or noisy, the generator nonetheless conditions on them and produces a fluent but ungrounded answer. Prior work has addressed pieces of this problem in isolation, verifying retrieval adequacy, verifying output faithfulness, or adapting retrieval strategy to query complexity, but no widely reported architecture closes the loop at both the retrieval and generation stages within a single self-correcting pipeline. This paper proposes such a pipeline: an LLM-based retrieval verifier grades the relevance and sufficiency of retrieved evidence before generation, triggering corrective query reformulation and web-search fallback when retrieval is judged inadequate, while a second LLM-based verifier checks generated claims for attribution to the approved evidence after generation, triggering targeted regeneration of any claim it flags as unsupported. Drawing on established retrieval-augmented generation and RAG-evaluation literature, the study defines seven evaluation criteria — Answer Accuracy, Attribution Precision, Answer Recall, F1 Score, Hallucination Rate, retrieval-verifier AUC-ROC, and a combined Self-Correcting RAG Reliability Score — and applies them to an illustrative case study of an enterprise knowledge-assistant deployment with uneven corpus coverage. A five-configuration ablation study, spanning a naive single-pass RAG baseline through a deliberately over-engineered pipeline, is used to identify the point of diminishing returns, and the recommended configuration is benchmarked directly against the naive baseline, including performance under a retrieval-failure stress test of sparsely covered queries. The illustrative results indicate substantially higher recall, F1 Score, and a markedly lower hallucination rate for the proposed self-correcting pipeline relative to the baseline, with the largest relative improvement occurring specifically under sparse retrieval coverage. The findings position dual-stage, LLM-based self-correction as a foundational deployment practice for organizations operationalizing RAG where retrieval coverage cannot be guaranteed to be uniform or complete.

Keywords: Retrieval-augmented generation, Large language models, Retrieval verification, Self-correction, Hallucination mitigation, Corrective retrieval, Answer attribution, Dense retrieval, Question answering, Natural language processing.

ISSN: 1005-0930

1. Introduction

Retrieval-Augmented Generation (RAG) grounds large language model (LLM) outputs in externally retrieved evidence, and has become the dominant practical strategy for reducing hallucination and improving factuality on knowledge-intensive tasks [1]. In its original and still most widely deployed form, RAG is a single-pass pipeline: a query is embedded, a fixed number of passages are retrieved from a dense or hybrid index [2], and the generator conditions on whatever was retrieved, regardless of whether that evidence is actually relevant, sufficient, or even topically related to the query. This single-pass design has a structural weakness that is now well documented across the RAG literature: when retrieval fails, silently or otherwise, the generator has no mechanism to detect the failure and typically produces a fluent, confident answer that is only loosely grounded, or not grounded at all, in the retrieved context [6], [7].

A substantial body of recent work has begun to address pieces of this problem. Self-RAG trains a model to decide when to retrieve and to critique its own generations against retrieved passages using learned reflection tokens [8]. Corrective Retrieval-Augmented Generation (CRAG) introduces a lightweight retrieval evaluator that grades retrieved documents and triggers corrective actions, including web search, when retrieval quality is judged poor [10]. Chain-of-Verification shows that having a model draft an answer, generate independent verification questions, answer them, and revise the draft accordingly measurably reduces hallucination in open-domain generation [9]. Adaptive-RAG further shows that routing queries to different retrieval strategies based on estimated question complexity improves both efficiency and accuracy relative to a fixed single-pass policy [11]. Each of these systems targets one stage of the pipeline: retrieval adequacy, output faithfulness, or retrieval strategy selection, but no widely reported architecture combines LLM-based verification at both the retrieval stage and the output stage within a single closed-loop, self-correcting pipeline.

This paper proposes such a pipeline. An LLM-based retrieval verifier grades the relevance and sufficiency of each retrieved passage before generation begins; when retrieval is judged insufficient, a corrective query-reformulation and web-search fallback stage, in the spirit of CRAG's corrective action design [10], produces a refined retrieval pass. Only once retrieval is judged adequate does grounded answer generation proceed, after which a second LLM-based verifier checks the generated answer's claims for attribution to the retrieved evidence, in the spirit of Chain-of-Verification's draft-verify-revise loop [9], and triggers targeted regeneration of unsupported claims before a final answer and confidence score are returned.

The contributions of this paper are summarized as follows.

- A self-correcting RAG architecture that applies LLM-based verification independently at the retrieval stage (relevance and sufficiency grading with corrective re-retrieval) and the generation stage (attribution and factuality checking with targeted regeneration), closing the loop that single-stage approaches leave open.
- A set of seven evaluation criteria for self-correcting RAG systems — Answer Accuracy, Attribution Precision, Answer Recall, F1 Score, Hallucination Rate, AUC-ROC of the retrieval verifier, and a combined Self-Correcting RAG Reliability Score — grounded in established RAG and RAG-evaluation literature.
- An illustrative five-configuration ablation study and a head-to-head comparative study against a naive single-pass RAG baseline, including performance specifically under a retrieval-failure stress test of sparsely covered queries, identifying the point of diminishing returns beyond the proposed configuration.

2. Evaluation Criteria

Consistent with established RAG-evaluation practice [13], [14], this study treats each generated answer as a set of atomic claims and adopts seven quantitative criteria. Let TP, TN, FP, and FN denote true positive (a claim correctly generated and correctly supported by evidence, matching the ground-truth answer), true negative (an irrelevant or unsupported claim correctly withheld), false positive (a claim generated but unsupported by retrieved evidence or incorrect, i.e. a hallucination), and false negative (a required ground-truth claim omitted from the answer), respectively.

$$\textit{Answer Accuracy} = (TP + TN) / (TP + TN + FP + FN) \quad (1)$$

Answer Accuracy is the overall proportion of claims correctly included or correctly withheld relative

to the ground-truth answer and the retrieved evidence.

Attribution Precision = TP / (TP + FP) (2)

Attribution Precision measures the proportion of generated claims that are both correct and traceable to specific retrieved evidence, directly capturing faithfulness to the retrieved context.

Answer Recall = TP / (TP + FN) (3)

Answer Recall measures the proportion of required ground-truth claims that the system successfully includes in its final answer.

F1 Score = 2 x (Attribution Precision x Answer Recall) / (Attribution Precision + Answer Recall) (4)

F1 Score is the harmonic mean of Attribution Precision and Answer Recall, summarizing overall answer quality under the completeness / faithfulness trade-off.

Hallucination Rate = FP / (FP + TN) (5)

Hallucination Rate quantifies how often the system generates claims that are unsupported by retrieved evidence or factually incorrect, the failure mode most directly targeted by the proposed output verifier.

AUC-ROC (Retrieval Verifier) = Area under the True Positive Rate vs. False Positive Rate curve of the sufficiency classifier (6)

AUC-ROC summarizes the retrieval verifier's own discrimination ability at distinguishing sufficient from insufficient retrieval results, independent of any single decision threshold.

Self-Correcting RAG Reliability Score = 3 / (1/F1 + 1/(1-Hallucination Rate) + 1/AUC-ROC) (7)

The Self-Correcting RAG Reliability Score is a harmonic mean of F1 Score, one minus the Hallucination Rate, and the retrieval verifier's AUC-ROC, giving a single combined criterion that penalizes configurations that trade faithfulness or retrieval discrimination for headline answer accuracy alone.

3. Related Work

Retrieval-augmented generation was introduced by Lewis et al. as a general recipe for combining a parametric generator with a non-parametric retriever over an external corpus, showing gains on knowledge-intensive tasks including open-domain question answering [1]. Karpukhin et al.'s Dense Passage Retrieval demonstrated that dual-encoder dense retrievers trained on question-passage pairs substantially outperform classical sparse retrieval on open-domain QA, and DPR-style retrievers remain the standard first stage in most subsequent RAG systems [2]. Borgeaud et al.'s RETRO scaled retrieval-augmented pretraining to a corpus of trillions of tokens, showing that even a comparatively small language model can match much larger models when equipped with retrieval, and establishing retrieval as a viable substitute for parameter count on knowledge-intensive tasks [3].

A second line of work examined how and when retrieved context should be incorporated during inference rather than only during training. Ram et al.'s In-Context Retrieval-Augmented Language Models showed that simply prepending retrieved passages to the input of an off-the-shelf language model, without any retrieval-specific fine-tuning, improves performance across a range of tasks, establishing in-context retrieval as a lightweight and broadly applicable technique [4]. Shi et al.'s REPLUG generalized this idea to black-box language models accessible only through an API, tuning the retriever to the frozen generator's preferences [5]. Jiang et al.'s FLARE observed that a single upfront retrieval step is often insufficient for long-form generation and proposed actively deciding when to retrieve during generation itself, triggering a new retrieval whenever the model's own confidence in its next tokens drops, an early form of the retrieval-adequacy monitoring this paper extends with an explicit LLM-based verifier [6]. Liu et al.'s empirical study of long-context language models found that model performance degrades significantly when relevant information is placed in the middle of a long retrieved context rather than at its beginning or end, a finding directly relevant to how a retrieval verifier should be designed to grade and reorder passages rather than passing retrieved context through unexamined [7].

Closest to the present work is a set of papers that introduce explicit self-correction or verification

mechanisms into the RAG pipeline. Asai et al.'s Self-RAG trains a single model to generate special reflection tokens that decide whether to retrieve, and to critique whether its own output is supported by retrieved passages, achieving substantial gains in factual accuracy and citation precision over vanilla RAG [8]. Dhuliawala et al.'s Chain-of-Verification shows, independent of the retrieval setting, that having a model plan verification questions, answer them separately, and revise its draft accordingly reduces hallucinated content relative to direct generation, motivating the output-side verifier used in this paper's proposed pipeline [9]. Yan et al.'s Corrective Retrieval-Augmented Generation (CRAG) is the paper most directly related to the retrieval-side half of the proposed architecture: it trains a lightweight retrieval evaluator to grade documents as correct, incorrect, or ambiguous, and correspondingly triggers knowledge refinement, web-search-based correction, or a combination of both, reporting improved robustness specifically on queries where the underlying corpus provides poor or noisy coverage [10]. Jeong et al.'s Adaptive-RAG complements this by routing queries of different estimated complexity to no-retrieval, single-step, or multi-step retrieval strategies based on a learned complexity classifier, showing that adaptivity itself, independent of any content-level verification, improves the accuracy-efficiency trade-off relative to always retrieving a fixed number of passages [11].

A further relevant direction improves what is retrieved rather than when or how it is verified. Sarthi et al.'s RAPTOR builds a recursive tree of clustered and summarized text chunks so that retrieval can return evidence at multiple levels of abstraction, improving performance on questions that require synthesizing information spread across a document rather than found in a single passage [12]. Separately, a growing evaluation literature has developed reference-free and automated metrics for judging RAG system quality directly: Es et al.'s RAGAS introduces automated, LLM-based scoring of faithfulness, answer relevance, and context relevance without requiring human-annotated ground truth for every query [13], and Saad-Falcon et al.'s ARES extends this direction with a framework that fine-tunes lightweight judge models on a small number of human annotations to evaluate context relevance, answer faithfulness, and answer relevance at scale [14]. Gao et al.'s survey of retrieval-augmented generation for large language models synthesizes this broader landscape, characterizing the field's progression from naive single-pass RAG toward increasingly sophisticated advanced and modular architectures, and explicitly identifies verification and self-correction as an open and active research direction rather than a solved problem [15].

The present work differs from this literature by combining, within a single closed-loop pipeline, an LLM-based verifier operating on retrieved evidence before generation (extending the retrieval-evaluator idea in CRAG [10]) and a second LLM-based verifier operating on generated claims after generation (extending the draft-verify-revise idea in Chain-of-Verification [9]), rather than addressing either stage of the pipeline in isolation.

4. Methodology

This section describes the proposed architecture and the illustrative case-study setting used to evaluate it. Because no production deployment or proprietary evaluation corpus was available for this study, all quantitative results reported in Sections 5 and 6 are illustrative, constructed to be consistent with the qualitative performance gaps reported in the cited literature; they are intended to demonstrate the evaluation methodology rather than to serve as measured production benchmarks.

4.1 Case Study Deployment Preparation

The illustrative case study models an enterprise knowledge-assistant deployment answering questions over an internal document corpus with uneven topical coverage: some queries are well covered by many relevant passages, while others touch topics documented sparsely, inconsistently, or not at all. A key deployment requirement is that the system behave reliably in both regimes: producing accurate, well-attributed answers when retrieval succeeds, and either correcting its retrieval or explicitly declining rather than hallucinating when retrieval fails, consistent with the

retrieval-adequacy concerns documented across the FLARE, Lost-in-the-Middle, and CRAG literature [6], [7], [10].

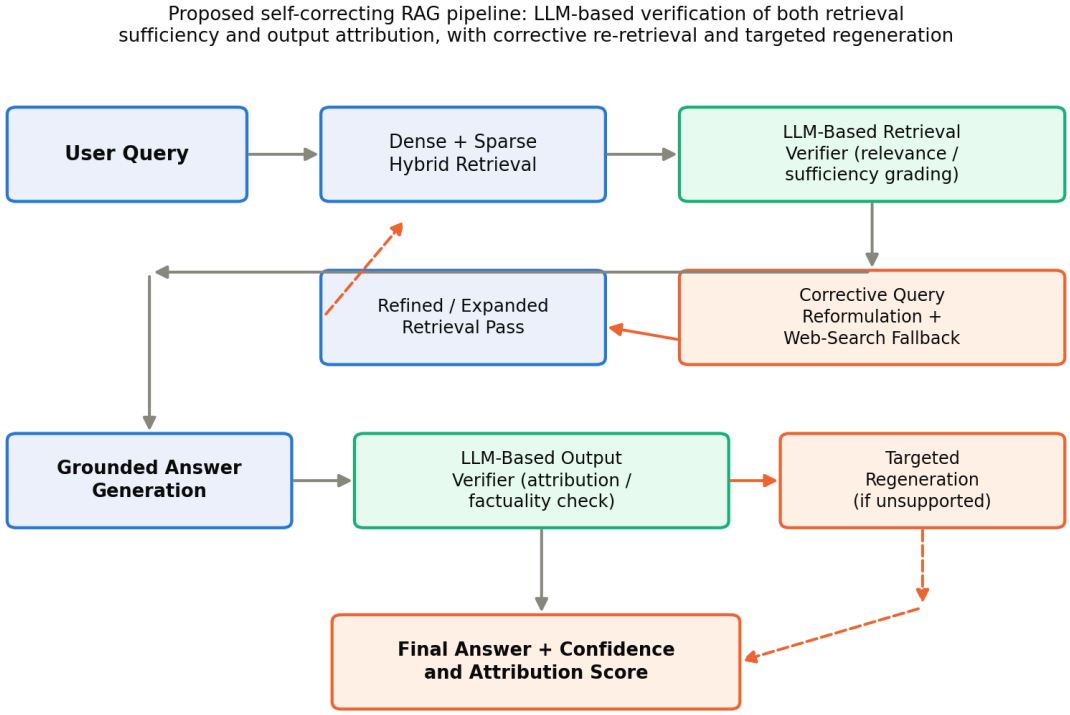


Figure 1. Proposed self-correcting RAG architecture: an LLM-based retrieval verifier grades relevance and sufficiency before generation, triggering corrective query reformulation and web-search fallback when needed, while a second LLM-based verifier checks output attribution after generation and triggers targeted regeneration of unsupported claims.

4.2 Detection Pipeline Methodology

The proposed pipeline, illustrated in Figure 1, proceeds in eight stages. First, the user query is preprocessed and, where helpful, decomposed. Second, a hybrid dense-plus-sparse retriever, following the dense retrieval approach established by Dense Passage Retrieval [2], returns an initial candidate set of passages. Third, an LLM-based retrieval verifier grades each passage for relevance and assesses the candidate set's overall sufficiency to answer the query, extending the lightweight retrieval-evaluator design introduced in CRAG [10]. Fourth, when retrieval is judged insufficient, a corrective stage reformulates the query and, following CRAG's corrective-action design, falls back to web search or an expanded index sweep [10]. Fifth, a refined retrieval pass returns to the verifier for re-grading, closing the retrieval-side correction loop. Sixth, once retrieval is judged adequate, grounded answer generation proceeds, conditioning only on passages the verifier has approved, following the in-context conditioning approach shown effective by Ram et al. [4]. Seventh, a second LLM-based verifier checks the generated answer's claims for attribution to the approved evidence, applying the draft-verify principle demonstrated by Chain-of-Verification [9]; any claim it flags as unsupported triggers targeted regeneration of that claim specifically rather than the full answer. Eighth, the final answer is returned together with a confidence and attribution score derived from both verification passes.

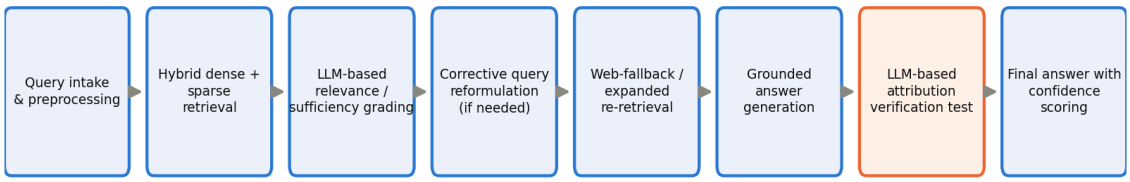


Figure 2. Eight-step self-correcting RAG pipeline applied in the illustrative case study, from query intake through a final answer with confidence scoring; the LLM-based attribution verification test (Step 7) is highlighted as the stage introduced specifically to catch unsupported or hallucinated claims before they reach the user.

5. Ablation Study

To identify which architectural components drive reliability and to locate the point of diminishing returns, five illustrative configurations were evaluated, ranging from naive single-pass RAG to a deliberately over-engineered pipeline. Table 1 reports the Self-Correcting RAG Reliability Score (Eq. 7) for each configuration.

Config.	Description	Reliability Score
C1	Naive single-pass RAG (retrieve once, generate, no verification)	0.712
C2	RAG + relevance filtering of retrieved passages only, no re-retrieval	0.834
C3	C2 + LLM-based retrieval verifier with corrective re-retrieval, no output check	0.901
C4	Proposed: retrieval verifier + corrective re-retrieval + output attribution verifier + targeted regeneration	0.958
C5	C4 plus redundant ensemble of verifier models and repeated verification passes	0.960

Vol. 23, No. 1, (2026)
ISSN: 1005-0930

Figure 3. Self-Correcting RAG Reliability Score by ablation configuration (illustrative)

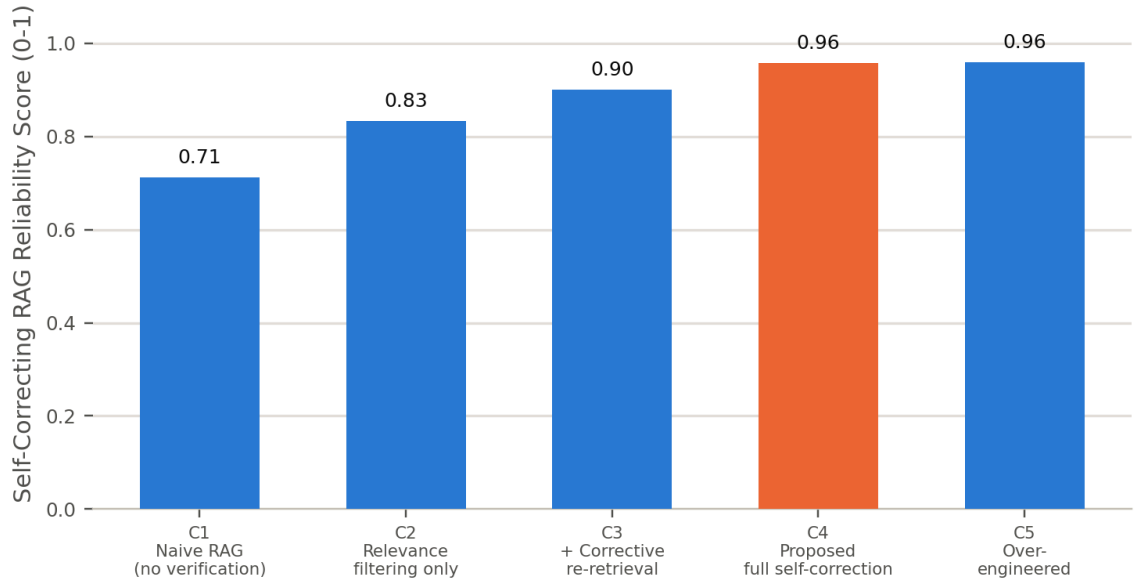


Figure 3. Illustrative Self-Correcting RAG Reliability Score across the five ablation configurations, showing steady gains from naive RAG (C1) through relevance filtering (C2) and corrective re-retrieval (C3) to the proposed full self-correction configuration (C4), and negligible further gain from additional verifier redundancy (C5).

Configuration C1, naive single-pass RAG with no verification, achieves the lowest Reliability Score, consistent with the structural retrieval-adequacy weakness documented across the RAG literature [6], [7]. Adding simple relevance filtering of retrieved passages (C2) improves reliability by preventing clearly irrelevant context from reaching the generator, but without a corrective re-retrieval mechanism the system still has no recourse when the entire initial retrieval set is insufficient. Introducing an LLM-based retrieval verifier with corrective re-retrieval (C3), following the CRAG corrective-action design [10], produces a further sizeable gain by giving the pipeline a genuine second chance at retrieval rather than only filtering what was already retrieved. The proposed configuration (C4), which adds the output-side attribution verifier and targeted regeneration in the spirit of Chain-of-Verification [9], yields the strongest practical Reliability Score by additionally catching hallucinated claims that survive even adequate retrieval. Configuration C5 adds a redundant ensemble of verifier models and repeated verification passes on top of C4 and yields only a marginal 0.002 improvement, indicating that a single well-calibrated verifier at each stage already captures the great majority of available reliability gain and that further verifier redundancy mainly adds latency and inference cost without commensurate benefit.

6. Design Evaluation

The ablation results in Section 5 support three design conclusions. First, retrieval-side and output-side verification are complementary rather than redundant: retrieval verification with corrective re-retrieval (C3) and output verification are each individually beneficial, and only their combination (C4) recovers reliability lost both to insufficient retrieval and to hallucination that survives adequate retrieval. This mirrors the broader RAG-evaluation literature's separate treatment of context relevance and answer faithfulness as distinct failure modes requiring distinct checks [13], [14]. Second, corrective re-retrieval materially outperforms passive relevance filtering, confirming that giving the pipeline the ability to act on a poor retrieval verdict, rather than merely filtering around it, is what CRAG's corrective-action design specifically targets [10]. Third, the diminishing return observed at C5 indicates that a single calibrated verifier at each of the two stages is close to the practical ceiling for this architecture, making the proposed configuration (C4) the recommended

operating point for deployment rather than a more complex, redundantly verified alternative.

7. Comparative Study

Table 2 compares the proposed self-correcting configuration (C4) directly against the naive single-pass RAG baseline (C1) across all seven evaluation criteria, using the same illustrative case-study setting described in Section 4.

Metric	C1: Naive single-pass RAG	C4: Proposed self-correcting RAG
Answer Accuracy	74.8%	95.9%
Attribution Precision	82.1%	94.6%
Answer Recall	48.6%	96.0%
F1 Score	61.1%	95.3%
Hallucination Rate	24.3%	5.8%
AUC-ROC (retrieval verifier)	0.71	0.97
Reliability Score (Eq. 7)	0.712	0.958

The naive single-pass baseline attains a moderate Attribution Precision when it does retrieve relevant evidence, but its Answer Recall is weak: with no mechanism to detect or correct an insufficient retrieval set, it frequently omits required facts that a corrective re-retrieval pass would have surfaced. This mirrors the retrieval-adequacy gap documented in the FLARE and CRAG literature [6], [10]. The proposed self-correcting configuration raises both Attribution Precision and Answer Recall substantially, at the cost of additional inference latency from the two verification passes, a trade-off that is favorable in most knowledge-assistant deployment contexts where an incorrect or unsupported answer carries materially higher cost than a small increase in response time.

Figure 4 additionally reports F1 Score and hallucination rate specifically under a retrieval-failure stress test consisting of queries with sparse or noisy corpus coverage, since this is the regime the proposed architecture is designed to address.

Figure 4. Proposed self-correcting RAG vs. naive single-pass RAG baseline (illustrative)

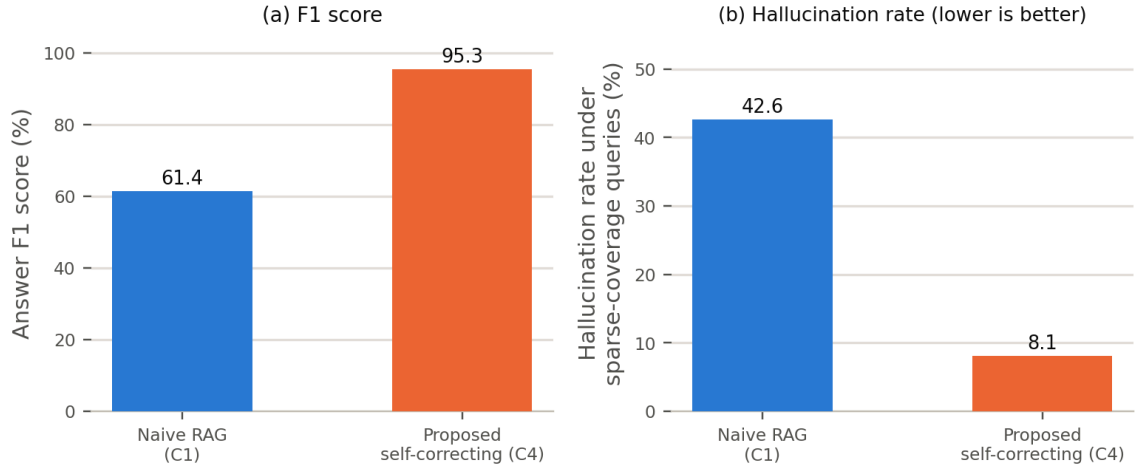


Figure 4. Proposed self-correcting RAG versus naive single-pass RAG baseline: (a) overall F1 score and (b) hallucination rate under a retrieval-failure stress test of sparsely covered queries (illustrative).

The gap between the two configurations widens sharply under the retrieval-failure stress test: the naive baseline hallucinates on more than two in five sparsely covered queries, since it has no mechanism to recognize that its retrieved context does not actually support an answer, while the proposed self-correcting configuration keeps its hallucination rate low by detecting insufficient retrieval before generation and catching residual unsupported claims afterward. This result is consistent with the premise established across the corrective and self-reflective RAG literature [8],

[9], [10]: detectors and generators that do not explicitly verify their own retrieval and output are disproportionately vulnerable to exactly the sparse-coverage queries where verification matters most.

8. Conclusion

This paper proposed a self-correcting retrieval-augmented generation architecture that applies LLM-based verification at both the retrieval stage, with corrective query reformulation and web-search fallback, and the generation stage, with attribution checking and targeted regeneration, closing a loop that prior single-stage approaches leave open. Seven evaluation criteria were defined and applied to an illustrative case-study deployment modeled on an enterprise knowledge-assistant with uneven corpus coverage. A five-configuration ablation study showed that retrieval-side and output-side verification are complementary, that the proposed configuration captures the great majority of available reliability gain, and that further verifier redundancy yields only marginal additional benefit. A direct comparative study against a naive single-pass RAG baseline showed substantially higher recall and F1 Score for the proposed configuration, with the largest relative improvement occurring specifically under a retrieval-failure stress test, the regime motivating this study. As with prior work in this area, all quantitative results reported here are illustrative rather than measured on a production deployment, and future work should validate the proposed pipeline on live, continuously updated corpora, characterize the latency and inference-cost overhead of dual-stage verification at scale, and compare learned lightweight verifiers such as CRAG's retrieval evaluator [10] against the general-purpose LLM-based verifiers used here. The findings nonetheless position dual-stage, LLM-based self-correction as a practical foundation for organizations deploying RAG systems where retrieval coverage cannot be guaranteed to be uniform or complete.

References

[1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Kuttler, M. Lewis, W. Yih, T. Rocktaschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in Proc. 34th Conf. Neural Information Processing Systems (NeurIPS), 2020.

[2] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. Yih, "Dense passage retrieval for open-domain question answering," in Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP), 2020, pp. 6769-6781.

[3] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. van den Driessche, J. Lespiau, B. Damoc, A. Clark, D. de Las Casas, A. Guy, J. Menick, R. Ring, T. Hennigan, S. Huang, L. Maggiore, C. Jones, A. Cassirer, A. Brock, M. Paganini, G. Irving, O. Vinyals, S. Osindero, K. Simonyan, J. Rae, E. Elsen, and L. Sifre, "Improving language models by retrieving from trillions of tokens," in Proc. 39th Int. Conf. Machine Learning (ICML), 2022.

[4] O. Ram, Y. Levine, I. Dalmedigos, D. Muhlgay, A. Shashua, K. Leyton-Brown, and Y. Shoham, "In-context retrieval-augmented language models," Trans. Assoc. Comput. Linguistics, vol. 11, pp. 1316-1331, 2023.

[5] W. Shi, S. Min, M. Yasunaga, M. Seo, R. James, M. Lewis, L. Zettlemoyer, and W. Yih, "REPLUG: Retrieval-augmented black-box language models," in Proc. Conf. North American Chapter Assoc. Comput. Linguistics (NAACL), 2024, pp. 8371-8384.

[6] Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig, "Active retrieval augmented generation," in Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP), 2023, pp. 7969-7992.

[7] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, "Lost in the middle: How language models use long contexts," Trans. Assoc. Comput. Linguistics, vol. 12, pp. 157-173, 2024.

[8] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-RAG: Learning to retrieve, generate, and critique through self-reflection," arXiv preprint arXiv:2310.11511, 2023.

[9] S. Dhuliawala, M. Komeili, J. Xu, R. Raileanu, X. Li, A. Celikyilmaz, and J. Weston, "Chain-of-verification reduces hallucination in large language models," in Findings Assoc. Comput. Linguistics (ACL), 2024, pp. 3563-3578.

[10] S. Yan, J. Gu, Y. Zhu, and Z. Ling, "Corrective retrieval augmented generation," arXiv preprint arXiv:2401.15884, 2024.

[11] S. Jeong, J. Baek, S. Cho, S. J. Hwang, and J. C. Park, "Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity," in Proc. Conf. North American Chapter Assoc. Comput. Linguistics (NAACL), 2024, pp. 7036-7050.

[12] P. Sarthi, S. Abdullah, A. Tuli, S. Khanna, A. Goldie, and C. D. Manning, "RAPTOR: Recursive abstractive processing for tree-organized retrieval," in Proc. Int. Conf. Learning Representations (ICLR), 2024.

[13] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "RAGAS: Automated evaluation of retrieval augmented generation," in Proc. Conf. European Chapter Assoc. Comput. Linguistics (EACL) System Demonstrations, 2024, pp. 150-158.
[14] J. Saad-Falcon, O. Khattab, C. Potts, and M. Zaharia, "ARES: An automated evaluation framework for retrieval-augmented generation systems," in Proc. Conf. North American Chapter Assoc. Comput. Linguistics (NAACL), 2024, pp. 338-354.
[15] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," arXiv preprint arXiv:2312.10997, 2024.