Agentic RAG with Self-Validation for Conversational Financial Document Analysis

Raghavendranath Kandula[1]

[1]Independent Researcher, United States

Abstract

Financial document question answering requires retrieving appropriate evidence, selecting financial quantities and executing calculations. We study a retrieval-augmented generation pipeline with BGE-M3 embeddings, FAISS retrieval and at most three generate–validate–refine attempts. The camera-ready evaluation uses documented hosted models and fixed public benchmark samples. On 50 FinQA questions, numerical reference agreement increases from 28/48 to 35/48 numerical items; on 50 ConvFinQA conversations, all-turn agreement changes from 132/178 to 134/178. On 50 FinanceBench questions, normalized literal reference agreement increases from 6/50 to 10/50. This last measure is a limited diagnostic for prose answers, not human-assessed accuracy. An additional FinanceBench control obtains 9/50 through three-pass self-revision without validator feedback, matching the same nine references as validated refinement with fixed evidence. The results support an auditable implementation and a concrete retrieval-recovery case, but do not establish a distinct factual-accuracy benefit of validation alone. Independent human evaluation remains future work.

Keywords

retrieval-augmented generation, financial question answering, iterative refinement, conversational question answering, reproducibility

1. Introduction

Financial reports combine tables, narrative explanations and period-dependent quantities. A correct calculation can still answer the wrong question if a retrieved value belongs to a different year, entity or financial concept. Retrieval-augmented generation (RAG) provides document context [1], while iterative tool use and self-feedback offer ways to revise an initial response [2, 3]. The practical question here is whether a bounded retrieve–generate–validate–refine workflow improves final reference agreement over a shared single-shot draft.

We retain this systems focus across FinanceBench, FinQA and ConvFinQA. Our contributions are an instrumented financial RAG workflow, a documented small-sample evaluation with model and data provenance, and component controls separating fixed-evidence revision from retrieval refinement. The camera-ready version reports new evaluations using the implementation, model settings and experimental records accompanying this work; these replace the earlier experimental summary. We make no claim to reproduce unavailable historical scores or establish state-of-the-art performance.

2. Related work and scope

RAG combines retrieval with generation [1]. ReAct interleaves reasoning with actions [2]; our workflow uses a bounded, explicit sequence of retrieval and checking calls rather than claiming a new general agent architecture. Self-Refine uses iterative self-feedback [3]. Self-RAG trains reflection-token behavior [4], whereas our generator and checker are prompted without additional training. Corrective RAG evaluates retrieval quality and can change retrieval actions [5]; our retrieval is confined to the benchmark-specified report or filing. These methods motivate the controls, but are not reimplemented baselines in this study.

Proceedings of the 5th International Workshop on Natural Language Processing for Knowledge Graph Construction (NLP4KGC 2026), co-located with the 25th International Semantic Web Conference (ISWC 2026), Bari, Italy, October 26, 2026

© 2026 Copyright for this paper by its author. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

published 2026-11-19

CEUR-WS.org/Vol-4335/paper9.pdf

FinanceBench evaluates open-book financial QA with source evidence and human assessment [6]. FinQA introduces financial numerical reasoning and executable programs [7]; ConvFinQA studies chains of numerical reasoning across dialogue turns [8]. We use these datasets for different retrieval settings, not interchangeable leaderboard tasks. RAGAs distinguishes retrieval and generation assessment [9]; we do not report a RAGAs faithfulness score or infer hallucination reduction from answer agreement.

A separate related manuscript by the author, *Error Propagation and Verification Limits in Conversational Financial Question Answering* (under review), studies controlled erroneous-history interventions, dependency prediction and offline verification/repair. The present work evaluates an online retrieval and refinement system across three benchmarks. It does not reuse that study's experimental records, intervention annotations, prompts, tables or figures. Shared public benchmarks and source/arithmetic checking ideas remain related ingredients; the checks here are system engineering components rather than a separately claimed verification contribution.

3. System and evaluated variants

Retrieval. We tokenize documents into 1,500-token chunks with 200-token overlap, preserving text through token-offset mapping. BGE-M3 [10] produces normalized 1,024-dimensional dense embeddings; FAISS [11] uses exact inner-product search (`IndexFlatIP`), with top $k = 10$. Retrieval is filtered to a known report for FinanceBench and known filing for FinQA/ConvFinQA. This metadata assistance limits applicability to unrestricted document search. Reference answers, gold programs and gold evidence-page selections are excluded from inference.

Generation, execution and refinement. A draft contains a value, explanation, chunk citations and, where applicable, a bounded arithmetic program. A checker using the same model examines financial target, period, units, source inputs and arithmetic. Rejected drafts receive operational feedback; the full arm may retrieve again using a suggested query. The loop stops on internal acceptance or after three answer attempts. The last attempt is scored; no reference-based best-of-attempt selection is used. Unknown and malformed outputs remain in the evaluation denominator. Arithmetic execution checks a program's result, not whether its operands or financial interpretation are correct.

Versioned benchmark adapters. The preserved FinQA and ConvFinQA adapters execute the model program and use its calculated value as the final scalar; the originally stated value is retained in the logs. Their same-model validation call sees the candidate and produces a source-audit program. Retrieval refinement appends the proposed query to the original question. FinanceBench uses a later typed adapter: number, Boolean, text or unknown, with explicit amount/scalar/percentage units. It preserves the declared final display, separately records raw calculation, and rejects inconsistent output contracts. A source-first audit sees only the question, report identifier and retrieved evidence; a separate comparison call then sees candidate and audit. The source-first audit excludes candidate and earlier feedback, but uses the same model and is not independent verification. A focused missing-fact query replaces the original search query. These differences preclude treating all three datasets as one identical validation implementation.

4. Experimental protocol

Samples and exposure. FinQA uses 50 questions from the official test split (48 numerical, two Boolean), one per filing, ordered by a seeded SHA-256 rule (seed 1820). Twenty development questions come from the development split; selected evaluation filings exclude previously observed development/evaluation filings. The retrieval corpus contains 107 public processed report pages for the selected filings. ConvFinQA uses 50 complete conversations from its public development split, selected by SHA-256 order (seed 1821), one per filing, excluding previously observed FinQA filings; five

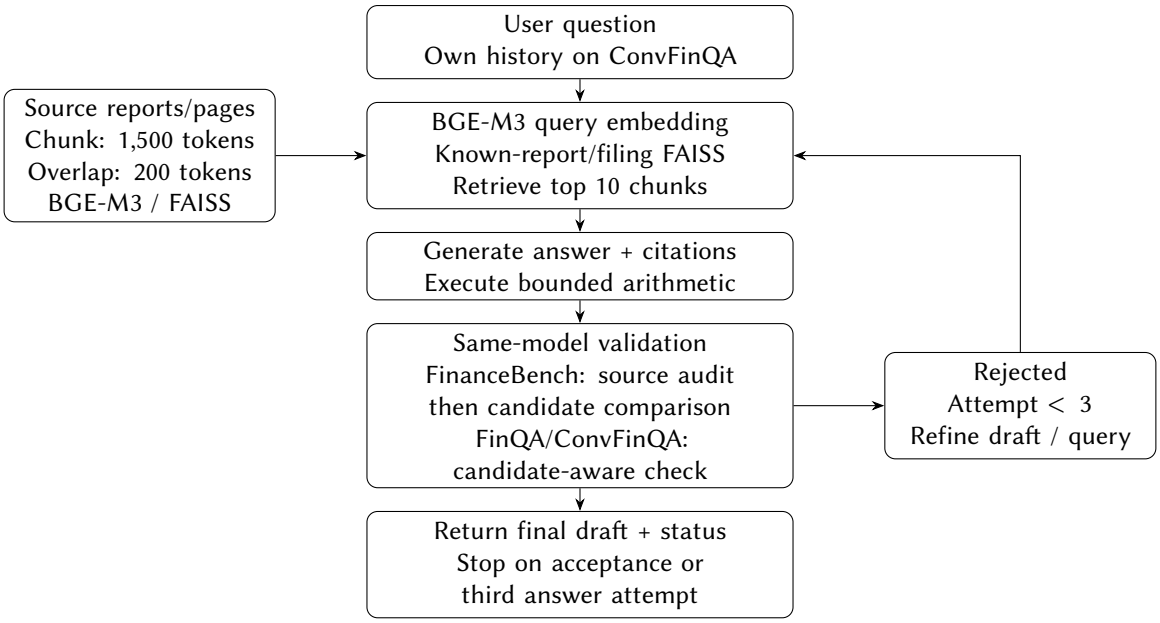


Figure 1: Full workflow; fixed-evidence controls retain the first retrieval. FinanceBench checking comprises source-first audit and candidate comparison; FinQA/ConvFinQA retain the earlier single-call checker. Internal acceptance is an operational state, not a factual-correctness label.

training-split conversations support development. Its corpus has 193 processed pages. We score all 178 evaluation turns and supply each arm with its own previous generated answers, never gold history or injected errors.

FinanceBench uses 50 fixed evaluation questions plus five separate development questions from the public 150-case release. The corpus comprises 44 complete source PDFs, 6,261 pages and 3,555 chunks, extracted with PyMuPDF 2.0 in sorted-text mode. The evaluation questions had already been inspected during earlier implementation audits. The final FinanceBench run and subsequent control are therefore post-hoc implementation evaluations, not untouched holdouts. Code and protocols were frozen before their respective requests, but that does not undo earlier sample exposure. Full IDs, source revisions, hashes and selection records accompany the experiments.

Models and comparators. DeepSeek `deepseek-v4.1-flash-1017` through OpenRouter/InferenceNet FP8 is the generator and same-model checker; Qwen `qwen3-32b` through DeepInfra FP8 is a single-shot comparator. Requests specify temperature 0, seed 18, maximum 4,096 output tokens and reasoning disabled. These hosted deployments replace the earlier self-hosted model description; seeds do not guarantee provider determinism. On FinQA and FinanceBench, single-shot and agentic arms share the exact initial draft and retrieved evidence. ConvFinQA uses arm-specific generated history and can have different later-turn initial drafts. Qwen uses the same initial evidence on FinanceBench/FinQA and its own generated dialogue history on ConvFinQA.

We also train an exploratory T5-small program generator [12]: 4,988 FinQA training examples after excluding 1,263 examples sharing selected development/evaluation filings, 100 development examples, and the same 50 evaluation questions. It uses three fixed epochs, batch 8, AdamW, learning rate 3×10^{-4}, seed 18, input/target limits 512/128 and greedy decoding. Its lexical top-ten-row reader differs from BGE retrieval; it is neither the unavailable historical "FinQA-T5" baseline nor a matched RAG ablation.

Metrics and uncertainty. FinQA *scalar numerical F1* is final-number agreement on the 48 numerical items. With one reference and one predicted scalar, it reduces to accuracy; it is not multi-number extraction F1 or symbolic program accuracy. Numerical comparison normalizes scalar formats and rounds to five decimal places, without sign or factor-100 forgiveness. Boolean agreement is included only in the separate 50-question total. ConvFinQA uses turn-weighted numerical agreement over all

178 turns; we additionally report the first up to three available turns (133 turns) and the fraction of fully correct conversations.

FinanceBench *normalized literal reference agreement* compares whole-scalar references numerically; otherwise it applies Unicode normalization, case folding, whitespace normalization and terminal-punctuation removal. Sixteen references are whole numerical scalars; 34 contain prose. Correct paraphrases or a short Boolean answer can disagree with a longer reference. This diagnostic does not implement the benchmark's human full-answer/reasoning rubric and cannot establish factual accuracy, source faithfulness or validator precision/recall. We retain every final output, including abstentions and internally unvalidated drafts. Paired 95% intervals resample filings (2,000 draws, seed 18); they express sample uncertainty, not immunity to sample exposure, metric bias or model contamination.

5. Results and component controls

Table 1
Final reference agreement on the documented samples. Each cell reports matches/denominator and rate. FB is a limited literal diagnostic; columns are different tasks/metrics. FinQA/ConvFinQA retain earlier adapters, so comparisons are within columns.

Method	FinanceBench	FinQA numerical	ConvFinQA all turns
DeepSeek single-shot	6/50 (.120)	28/48 (.583)	132/178 (.742)
DeepSeek full agentic	10/50 (.200)	35/48 (.729)	134/178 (.753)
Qwen3-32B single-shot	8/50 (.160)	28/48 (.583)	114/178 (.640)

FinQA's agentic arm changes seven numerical reference mismatches to matches and no matches to mismatches. The paired numerical difference is .146 (95% filing interval [.063, .250]); including the two Boolean items gives 37/50 versus 30/50. The exploratory T5-small baseline obtains 12/48 numerical matches (.250), and 12/50 overall (.240). Four evaluation inputs are truncated; retrieval, model size and training differ substantially from the hosted models. This result does not establish superiority over a published T5 system.

ConvFinQA's end-to-end difference is .011 (interval [−.010, .035]), with three mismatches becoming matches and one reverse change. The first-up-to-three-turn agreement is 105/133 (.789) versus 103/133 (.774); completely correct conversations are 26/50 versus 24/50. Because arms carry different generated histories, differences combine current-turn refinement and earlier-answer effects. The small uncertain change does not support a general claim that validation resolves conversational errors.

Table 2
FinanceBench controls on the same 50 questions and shared first drafts. Calls are unique attributed generation/checking requests, including reused initial calls. The three-pass control always scores pass three. Checking arms can stop early and use more total calls; compute is not matched.

Method	Matches	Mean answer passes	Calls	Invalid finals
Single-shot	6/50	1.00	50	6
Three-pass generic self-revision	9/50	3.00	150	5
Validated, fixed evidence	9/50	1.96	231	1
Full retrieval + validation	10/50	1.88	240	1

FinanceBench full-minus-single literal agreement is .080 (interval [.018, .170]), a descriptive exposed-sample result. The additional self-revision control receives only its previous draft, a constant reconsideration instruction and pass number. It receives no validator verdict, source audit, calculator result, parser diagnosis or reference answer, and never changes evidence. Both fixed-evidence multi-pass arms match exactly the same nine references. Thus explicit validation adds no literal-reference matches over generic revision in this sample. The validated arm has fewer invalid final output contracts (one

versus five), which is an execution/format distinction, not an independent financial-correctness finding. Full refinement adds one match over either control, Adobe question 04735; the full-minus-control difference .020 has interval [.000, .068]. Query reformulation, repeated retrieval and extra checking calls are not individually isolated. Python execution is not separately ablated; live EDGAR lookup is not used.

6. Worked trace, runtime and limitations

Adobe 2015: operating cash flow / total current liabilities (two decimals).

1. Evidence gap. Initial retrieval supplies operating cash flow but lacks total current liabilities, preventing calculation of the requested ratio.

↓ **2. Focused retrieval query:** "Adobe 2015 10-K consolidated balance sheets total current liabilities as of November 27, 2015".

↓ **3. New retrieved inputs:** operating cash flow 1,469,502 (thousand dollars; chunk token offset 57,200); total current liabilities 2,213,556 (same units; offset 54,600). Both chunks belong to `ADOBE_2015_10K/report`.

↓ **4. Final calculation:** 1,469,502/2,213,556 = 0.66386484 ...; declared answer **0.66**. It passes the operational checks and matches the reference. The fixed-evidence controls do not acquire the missing liability evidence.

Figure 2: An observed retrieval-recovery trajectory, summarized from saved requests and retrieved chunks. Numbers are reported model/source inputs, not a new human-labelled evaluation. This single success is illustrative, not general performance evidence.

The trace in Figure 2 demonstrates the intended online loop. Failures remain: for General Mills question 04103, the final cash-conversion-cycle program contains forward arithmetic references and is rejected by the executor despite a favorable model comparison. A model can also select the wrong financial formula while remaining arithmetically consistent. Operational success therefore cannot certify a financial answer.

Latency depends on the adapter and hosted service. FinQA averages 1.32 answer attempts and 4.20 stored API seconds per agentic trajectory versus 1.42 seconds single-shot; these sum saved successful-call durations and exclude retrieval, embedding, queueing and interrupted transport. They are not end-to-end controlled hardware measurements. FinanceBench full refinement averages 1.88 answer attempts but attributes 240 calls versus 50 single-shot calls, with source auditing and comparison adding work. Exact-payload caching shares requests across arms; call totals cannot be summed as separately billed experiments. Embeddings/execution run locally on macOS ARM64; generator hardware is provider-managed. We make no A100, GPTQ memory, offline deployment or two-second retry claim.

Sorted PDF text can lose table headers, continuation rows and sign/column alignment; processed benchmark pages are also easier than raw reports in some respects. Chunking and executable arithmetic cannot repair a wrongly extracted operand. We have not conducted a systematic extraction robustness study, nor evaluated an additional random EDGAR corpus or alternate embedding model. Small selected public samples, public training contamination, known-document filtering, versioned adapters, sample exposure and unmatched call/stopping budgets restrict generalization.

Human evaluation remains future work. Generation and validation use the same model and may share financial-interpretation errors. No independent human audit of generated answers or validator decisions has been conducted. Internal acceptance and benchmark reference agreement do not establish factual correctness or claim-level faithfulness. Future work includes source-aware human annotation of complete answers and validator decisions to estimate precision, recall, false acceptance and false rejection. We do not report an empirical faithfulness score, hallucination-reduction percentage or correctness guarantee.

7. Reproducibility and conclusion

Code, configurations and versioned result summaries are publicly available at https://github.com/raghavendranath/agentic-financial-rag. The experimental records retain benchmark/source revisions, selected IDs, input/index/code hashes, prompts, provider settings, complete drafts, retrieved chunks, arithmetic results and frozen scorers. Saved requests support offline replay without an API key; live runs require a separate local credential. The FinanceBench implementation passes 39 regression tests and the isolated self-revision control passes 12 tests. These verify software behavior and record integrity, not independent financial correctness. The optional T5 experiment has its own training protocol and checkpoint provenance.

The study provides a bounded, auditable financial RAG implementation. Final scalar agreement improves on the selected FinQA sample; the conversational gain is small and uncertain. The FinanceBench control shows that generic revision can match validated fixed-evidence refinement, while a focused retrieval change recovers one additional reference. Stronger claims require independent full-answer assessment, fresh evaluation samples and controls matching total calls and stopping behavior.

Declaration on Generative AI

OpenAI Codex assisted with software development and testing, analysis of experimental outputs, and manuscript editing and formatting. DeepSeek and Qwen are the experimental models described in Section 4. The author is responsible for the content.

References

[1] P. Lewis et al., Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, NeurIPS 33 (2020) 9459–9474. https://arxiv.org/abs/2005.11401.

[2] S. Yao et al., ReAct: Synergizing Reasoning and Acting in Language Models, ICLR (2023). https://arxiv.org/abs/2210.03629.

[3] A. Madaan et al., Self-Refine: Iterative Refinement with Self-Feedback, NeurIPS 36 (2023). https://arxiv.org/abs/2303.17651.

[4] A. Asai, Z. Wu, Y. Wang, A. Sil, H. Hajishirzi, Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection, ICLR (2024). https://arxiv.org/abs/2310.11511.

[5] S.-Q. Yan, J.-C. Gu, Y. Zhu, Z.-H. Ling, Corrective Retrieval Augmented Generation (2024). https://arxiv.org/abs/2401.15884.

[6] P. Islam, A. Kannappan, D. Kiela, R. Qian, N. Scherrer, B. Vidgen, FinanceBench: A New Benchmark for Financial Question Answering (2023). https://arxiv.org/abs/2311.11944.

[7] Z. Chen et al., FinQA: A Dataset of Numerical Reasoning over Financial Data, EMNLP (2021) 3697–3711. https://doi.org/10.18653/v1/2021.emnlp-main.300.

[8] Z. Chen, S. Li, C. Smiley, Z. Ma, S. Shah, W. Y. Wang, ConvFinQA: Exploring the Chain of Numerical Reasoning in Conversational Finance Question Answering, EMNLP (2022) 6279–6292. https://doi.org/10.18653/v1/2022.emnlp-main.421.

[9] S. Es, J. James, L. Espinosa Anke, S. Schockaert, RAGAs: Automated Evaluation of Retrieval Augmented Generation, EACL System Demonstrations (2024) 150–158. https://doi.org/10.18653/v1/2024.eacl-demo.16.

[10] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, Z. Liu, M3-Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation (2024). https://arxiv.org/abs/2402.03216.

[11] J. Johnson, M. Douze, H. Jégou, Billion-scale similarity search with GPUs, IEEE Transactions on Big Data 7 (2021) 535–547. https://doi.org/10.1109/TBDATA.2019.2921572.

[12] C. Raffel et al., Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, JMLR 21(140) (2020) 1–67. https://jmlr.org/papers/v21/20-074.html.