DO NOT BLAME YOUR RAG SYSTEM: THE MULTI-HOP QA BENCHMARKS NEED REVISION

Anonymous authors
Paper under double-blind review

ABSTRACT

Retrieval-augmented generation (RAG) in modern agents underpins knowledge, skill, memory and tool retrieval. RAG systems are primarily evaluated on three multi-hop question answering (QA) benchmarks, namely HotpotQA, 2WikiMultiHopQA and MuSiQue, which test whether a system can retrieve the relevant documents and generate the correct answer by reasoning over them. However, we show that these benchmarks have significant flaws. On average, **39.8**% of the samples carry at least one defect; for example, the question cannot be answered as written, the gold documents do not support the official answer, or the official answer is wrong. Our preliminary studies further show that even when given the gold documents, models of different sizes and families still miss the official answer on **15.9**% to **24.8**% of all questions, mainly on the defective samples. These defects inherently bias the evaluation of RAG systems, which rests on a flawed standard that requires urgent revision. We target this goal and release two audited versions of each benchmark: a repaired version (22,255 samples in total) based on our LLM auditing framework and a human-verified subset (2,732 samples in total). Notably, by auditing all 22,398 samples in the development sets of the three benchmarks, our LLM Auditor flags 35.6% of HotpotQA, 26.8% of 2WikiMultiHopQA and 57.1% of MuSiQue as defective under a taxonomy of twelve defect labels we define. We further evaluate the state-of-the-art RAG methods on our audited benchmarks to set up a reliable baseline for the community. We hope these released benchmarks and our framework will enable more reliable evaluation and development of future RAG systems.

1 INTRODUCTION

Retrieval-augmented generation (RAG) helps large language models (LLMs) generate accurate answers from retrieved external knowledge. LLM-based agents now use RAG to retrieve documents, skills (Wang et al., 2024; Qin et al., 2024), memory context (Packer et al., 2023; Gutiérrez et al., 2025) and Model Context Protocol (MCP) services (Hou et al., 2026; Gan & Sun, 2025). A RAG system typically retrieves documents relevant to a question from a large corpus and then uses them to generate an answer (Lewis et al., 2020). When the question requires it, the system reasons across those documents in multiple steps (Gao et al., 2023). As summarized in Table 1, researchers use multi-hop question answering (QA) to evaluate RAG systems, including agentic search methods (Li et al., 2025; Jin et al., 2025a; Chen et al., 2025). HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020) and MuSiQue (Trivedi et al., 2022) are the most widely used benchmarks for this task. Many recent RAG methods report results on these benchmarks' development

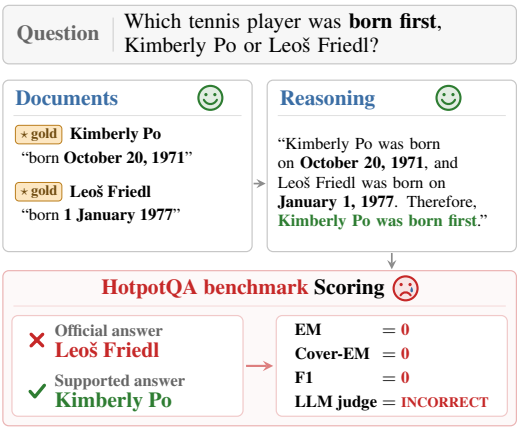


Figure 1: A sample of incorrect official answer in HotpotQA.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: Sixteen recent RAG methods and the benchmarks they evaluate on. Hot, 2Wi and MSQ denote HotpotQA, 2WikiMultiHopQA and MuSiQue.

Work	Venue	Hot	2Wi	MSQ	Work	Venue	Hot	2Wi	MSQ
IRCoT (Trivedi et al., 2023)	ACL 23	✓	✓	✓	Chain-of-Retrieval (Wang et al., 2025)	NeurIPS 25	✓	✓	✓
SiReRAG (Zhang et al., 2025b)	ICLR 25	✓	✓	✓	Search-R1 (Jin et al., 2025a)	COLM 25	✓	✓	✓
Search-o1 (Li et al., 2025)	EMNLP 25	✓	✓	✓	FrugalRAG (Java et al., 2026)	ICLR 26	✓	✓	✓
HippoRAG 2 (Gutiérrez et al., 2025)	ICML 25	✓	✓	✓	HiPRAG (Wu et al., 2026b)	ICLR 26	✓	✓	✓
LogicRAG (Chen et al., 2026a)	AAAI 26	✓	✓	✓	A^2Search (Zhang et al., 2026)	ICLR 26	✓	✓	✓
LinearRAG (Zhuang et al., 2026)	ICLR 26	✓	✓	✓	Interact-RAG (Hui et al., 2026)	ICLR 26	✓	✓	✓
STRIDE (Chen et al., 2026b)	SIGIR 26	✓	✓	✓	TIPS (Xie et al., 2026)	ICLR 26	✓	✓	✓
MemGraphRAG (Wu et al., 2026a)	KDD 26	✓	✓	✓	Cycle-Consistent Search (An et al., 2026)	COLM 26	✓	✓	✓

sets (their validation splits), and some also train on their training sets (Wang et al., 2025; Java et al., 2026; Jin et al., 2025a).

However, our extensive audit reveals significant flaws in samples from all these benchmarks (Appendix A.5). A concrete example from HotpotQA is shown in Figure 1 (full sample in Appendix A.3). The question asks which of two tennis players was born first. The corresponding *gold* documents give both birth dates, so *Kimberly Po* is the correct answer. Yet the benchmark wrongly gives *Leoš Friedl*, who was born in 1977, as the official answer.

The incorrect official answer in the above example is only one of the twelve defect labels in our taxonomy, identified by our extensive analysis on the benchmarks (Section 4.1). Across all 22,398 samples in the three public development sets, our auditing pipeline flags 35.6% of HotpotQA, 26.8% of 2WikiMultiHopQA and 57.1% of MuSiQue as defective. These identified defective samples can understate reported performance, distort comparisons between methods and introduce misleading training signals when benchmark feedback is used for learning (Jin et al., 2025a; Chen et al., 2025; Wu et al., 2026b), and can even be disastrous for modern recursive self-improving RAG systems (Duan et al., 2026).

However, to the best of our knowledge, no prior work has systematically studied these defects and proposed a correction in these widely adopted multi-hop QA benchmarks at full scale for the public benefit. Given the total of 22,398 samples, we build an LLM auditing framework to scale the analysis and propose a repair when a sample is identified as defective. We further employ six human annotators to cross-review the audits and proposed repairs on 1,000 random samples from each benchmark over two rounds. Ultimately, we release two cleaner versions, the *repaired* benchmarks (22,255 samples), which apply these repairs to the full development sets, and the *verified* benchmark (2,732 samples), which keeps only the samples that pass both rounds of human review.

Our main contributions are as the following:

- **Revisiting.** We conduct a manual audit and an Oracle experiment to show that defects are widespread across all three benchmarks and account for many failures even when models are given the gold documents. We distill these defects into a taxonomy of twelve clearly defined labels, drawn from an LLM analysis of the full development sets and refined by hand.
- **Repairing.** We develop a semi-automatic audit and repair framework that combines an LLM Auditor with cross-checking by human annotators. We release two cleaner versions of each benchmark, the *repaired* benchmarks and the *verified* benchmark. Together, the two versions provide the research community with a cleaner basis for evaluating RAG methods.
- **Re-evaluating.** We re-evaluate nine state-of-the-art RAG methods on the original, repaired and verified benchmarks. The original benchmarks understate every method's performance by different amounts, and can erase one method's apparent lead over another.

2 Defects of Multi-Hop QA Benchmarks

A RAG system relies mainly on two components, retrieval and generation, and its accuracy is bounded by both (Lewis et al., 2020; Gao et al., 2023). Figure 1 raises the question of how much of the shortfall reported on these benchmarks belongs to the benchmarks rather than to the systems. To separate the two, we run an *Oracle* experiment that removes retrieval altogether and hands the model the question together with its gold documents. Retrieval can no longer fail, so a wrong answer must come either from the model's reading of the documents or from the sample itself. We run it on

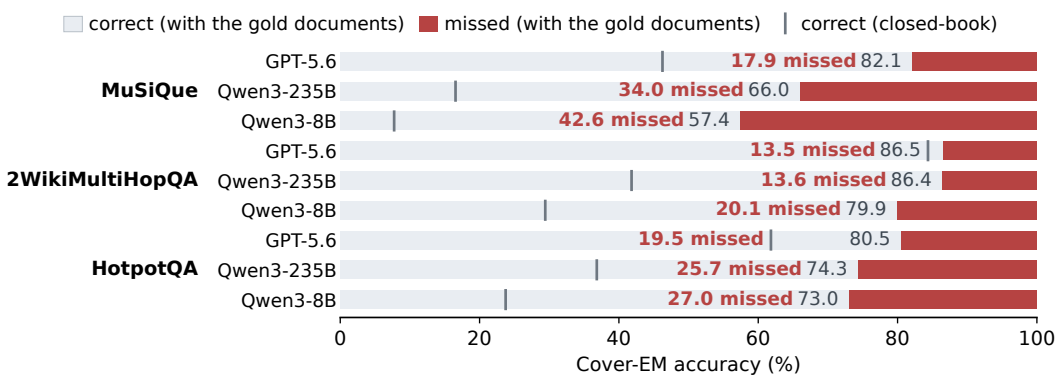


Figure 2: Even perfect retrieval fails. Models given gold documents still miss many official answers.

all 22,398 samples with three backbones, Qwen3-8B and Qwen3-235B as open-weight models and GPT-5.6 as a frontier proprietary one, alongside a closed-book control that sees no documents, and score the answers by Cover-EM. The full setup and all four metrics are in Appendices A.4 and F.3.

Even with the gold documents handed over, the models still miss the official answer on many questions (Figure 2). This holds for every model, regardless of its size, from Qwen3-8B to Qwen3-235B. It also holds regardless of model family, open-weight or proprietary, since GPT-5.6 still misses 13.5% to 19.5% of the questions. A stronger model narrows the gap but does not close it.

We are not the first to observe this shortfall. PRISM (Nahid & Rafiei, 2026) and DEXTER (V et al., 2024) run the same Oracle setting on samples of the benchmarks, and both read the errors that remain with the gold evidence in hand as failures of the model's reasoning (Appendix A.4).

To see what lies behind these failures, we audit 100 random samples from each benchmark by hand (Appendix A.5). Many of them are defective in themselves; for example, the question cannot be answered as posed, the gold documents do not support the official answer, or the official answer is wrong. Such defects affect nearly two thirds of MuSiQue, about a third of HotpotQA and a quarter of 2WikiMultiHopQA, and Section 6.3 shows that the models' failures concentrate on the defective samples. The shortfall is therefore *largely set by defects in the benchmarks themselves*, not only by the model's limited ability to understand the gold documents and generate the answer. Prior work thus tells only part of the story, and this argument motivates the rest of this work.

3 Related Work

Related work on multi-hop QA benchmarks. Platinum Benchmarks (Vendrow et al., 2025) cleans subsets of fifteen benchmarks spanning mathematics, logic, table understanding, reading comprehension, commonsense reasoning and visual understanding. Of the three benchmarks we study, Platinum covers only HotpotQA, through a 250-question sample. It asks how reliable frontier LLMs are across task types, so a small clean set from each domain is all it needs. We ask a different question, whether the benchmarks the RAG field reports on are sound, and that difference changes both what we inspect and what we do with a defect. Platinum inspects only what models get wrong, reviewing by hand every example that at least one frontier LLM answers incorrectly. Our work instead uses a semi-automatic framework, which scales more easily than manual review. The LLM Auditor audits all 22,398 samples and human annotators verify 1,000 per benchmark, whether or not a model answers them correctly. We repair the defects we find rather than dropping the samples that carry them. SAFE (Kwon et al., 2026) filters the same three benchmarks for unverifiable supervision while training a step verifier, discarding defective samples rather than repairing them.

Related work on benchmarks in other domains. Northcutt et al. (2021) audit the test sets of ten vision, text and audio benchmarks for label errors and release the human-validated errors for all ten, with a corrected label wherever reviewers agree on one. MMLU-Redux (Gema et al., 2025) audits MMLU under an error taxonomy and releases manually re-annotated questions covering all 57 subjects. KG-RAG Pitfalls (Zhang et al., 2025a) audits sixteen knowledge-graph QA benchmarks by

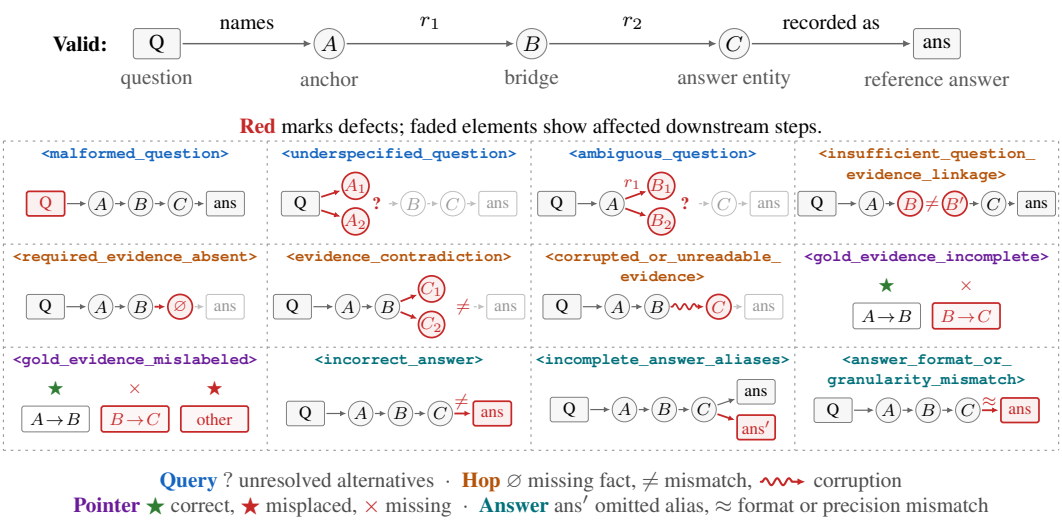


Figure 3: 12 defect labels (four families) on a shared illustrative reasoning chain.

hand and builds a new benchmark instead of repairing the old ones. MQuAKE-Remastered (Zhong et al., 2025) audits the multi-hop knowledge-editing benchmark MQuAKE, repairs the clerical and procedural errors it finds rather than discarding the affected cases, and re-benchmarks existing methods. Most of its checks can be automated because every case is generated from a Wikidata triple chain against which question and answer can be verified. The three benchmarks we audit have no such ground truth; the reasoning chains that 2WikiMultiHopQA and MuSiQue supply were never checked for whether the passages they point to state them (Section 5.1), so every judgment has to be made from the documents themselves. Across these domains, auditing a widely used benchmark and releasing what the audit finds, whether error annotations, corrected labels, a repaired dataset or a replacement benchmark, is established practice. Multi-hop QA for RAG has not had that treatment at full scale, and our work fills this gap. Appendix B discusses these works in detail.

4 A Framework for Auditing and Repairing Benchmarks

4.1 A Taxonomy of Twelve Defect Labels

We introduce the twelve defect labels, grouped into four families (Query, Hop, Pointer, Answer). We first distill them from an exploratory GPT-5.6 analysis of the full development sets and then refine their definitions and boundaries by hand on 300 random samples (Appendix A.5). These twelve labels are shared by the human annotators and the LLM Auditor. We read a two-hop sample as one chain that runs from the question through the anchor A and the bridge B to the answer entity C, following the relations r_1 and r_2. Figure 3 draws, for every label, where this chain breaks. Appendix C gives the definition and boundary rules of each label, with a schematic and an example.

Query family. The defect sits in the wording of the question. `<malformed_question>` has text broken enough to obscure what is being asked. `<underspecified_question>` names an anchor that matches more than one entity in the documents. `<ambiguous_question>` fixes its anchor but follows a relation that yields more than one supported value. The fork is what separates the two: it sits at the anchor when the question is underspecified and at the relation when it is ambiguous.

Hop family. The chain cannot be completed inside the documents. `<insufficient_question_evidence_linkage>` means that each hop is stated but the documents show the two ends of the bridge, B and B', to be different entities that merely share a name. `<required_evidence_absent>` means that no supplied document states the needed fact. `<evidence_contradiction>` means that the documents give two values that cannot both hold. `<corrupted_or_unreadable_evidence>` means that the text is destroyed exactly where the fact should stand.

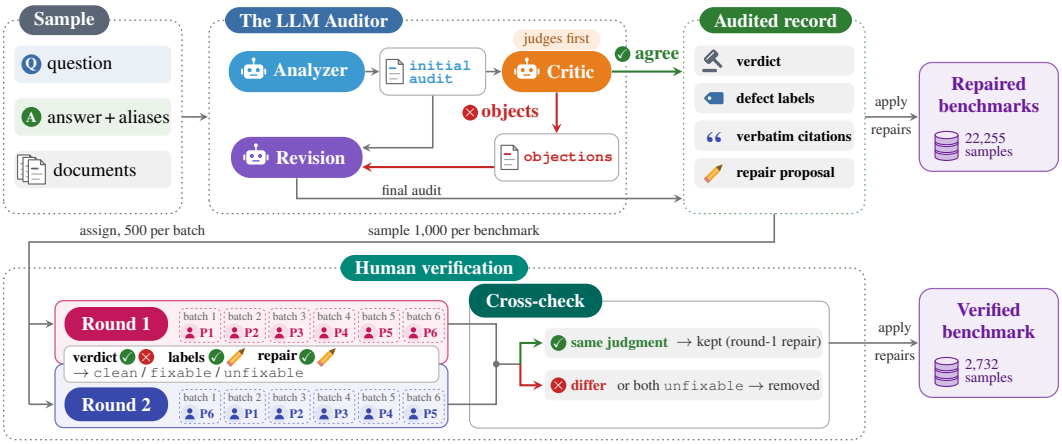


Figure 4: The semi-automatic audit and repair framework. Top: the LLM Auditor. Bottom: human verification, where each of the six batches is judged by two annotators from P1–P6, one per round.

Pointer family. The documents together can answer the question, but the text carrying the official gold marks cannot. `<gold_evidence_incomplete>` means that every mark is useful and a needed one is missing, so the repair only adds marks. `<gold_evidence_mislabeled>` means that a mark sits on text the chain never uses while the needed text is unmarked, so the repair also removes marks.

Answer family. The chain is healthy and the defect sits after it, in what the benchmark wrote down. `<incorrect_answer>` means that the official answer names a different thing from the one the documents support. Two spellings of the same thing are never incorrect. `<incomplete_answer_aliases>` means that the alias list rejects a valid form of the right answer. `<answer_format_or_granularity_mismatch>` means that the benchmark records the right thing at the wrong precision, or accepts a form that does not answer the question.

4.2 The Semi-Automatic Audit and Repair Framework

Building on the taxonomy, we build the semi-automatic audit and repair framework, which examines every sample of the three benchmarks. It has two parts, the LLM Auditor (Figure 4, top) and human verification (Figure 4, bottom). The LLM Auditor audits every sample first, and human annotators then verify its audits on 1,000 samples per benchmark, which is what makes the audit semi-automatic. The two parts complement each other. The LLM Auditor lets the framework scale to every sample while keeping human effort low, and human verification, in which two annotators cross-check each sample, makes the framework more trustworthy and yields higher-quality samples.

4.2.1 The LLM Auditor

The LLM Auditor applies the taxonomy of Section 4.1 over the original benchmarks in full, all 22,398 samples. Given a sample, it names every label of Section 4.1 that applies, quotes the text behind each claim and proposes a repair when the sample is defective. It does so in at most three LLM calls, in an order fixed in code. An analyzer drafts an initial audit, a critic judges the sample on its own before it reads that audit and then either agrees with it or objects, and only when the critic objects does a revision call write the final audit (Figure 4). Judging first is what keeps the critic from anchoring on the analyzer's conclusion. The verdict of a sample is derived in code from the labels in its final audit, never written by a model. The analyzer, the critic and the revision run on GPT-5.6 at high reasoning effort. Appendix D gives what each of the three calls receives and does, their setup and retry policy, the prompts and the automatic checks applied to every audit.

4.2.2 Human Verification Protocol

With all three benchmarks now audited and repaired by the LLM Auditor, we draw 1,000 samples at random from each benchmark, 3,000 in all, for human review. A thousand samples per benchmark

		HotpotQA	2WikiMultiHopQA	MuSiQue
Query	<malformed_question>	1.0	0	1.6
	<underspecified_question>	0.6	1.0	**5.4**
	<ambiguous_question>	**2.9**	**3.0**	**5.8**
Hop	<insufficient_question_evidence_linkage>	0.2	0.2	1.3
	<required_evidence_absent>	**8.5**	**15.7**	**39.7**
	<evidence_contradiction>	<0.1	<0.1	0.2
	<corrupted_or_unreadable_evidence>	<0.1	0.1	<0.1
Pointer	<gold_evidence_incomplete>	**4.9**	1.5	0.8
	<gold_evidence_mislabeled>	1.3	**2.4**	1.1
Answer	<incorrect_answer>	**4.3**	**4.2**	**8.4**
	<incomplete_answer_aliases>	**15.9**	2.0	**4.2**
	<answer_format_or_granularity_mismatch>	**2.8**	2.0	**4.5**
Any defect		**35.6**	**26.8**	**57.1**

Figure 5: Defect distribution of the original benchmarks. Each cell gives the percentage of that benchmark's samples carrying the label. A sample may carry several labels.

is the size on which many recent RAG methods evaluate (Appendix E.2), so the verified benchmark is drawn at the field's standard scale. Six annotators carry out the work, and every sample is judged twice, by two different annotators. In the first round, the 3,000 samples are split into six batches of 500 and one batch goes to each annotator. In the second round, each batch goes to an annotator other than its first one. In both rounds the annotator does the same work, starting from the LLM Auditor's final audit of each sample, confirming or overturning its verdict, correcting its labels, accepting or rewriting its repair and finally judging the sample `clean`, `fixable` or `unfixable`. The two annotators of each sample work independently, and the second does not see the first one's judgment, so the second round cross-checks the first. A sample on which the two annotators disagree on this judgment is disputed and removed, as is a sample that both judge `unfixable`. Every other sample enters the verified benchmark, and a `fixable` one keeps the repair of its first round. Appendix E gives the sampling procedure, the evaluation sample sizes reported in prior work and the annotation protocol in full, and Appendix E.4 the agreement between the two annotators.

5 Auditing the Benchmarks at Scale

5.1 Defect Distribution of the Original Benchmarks

Figure 5 shows the defect distribution of the original benchmarks as audited by the LLM Auditor. The LLM Auditor marks 35.6% of HotpotQA, 26.8% of 2WikiMultiHopQA and 57.1% of MuSiQue as defective. Each distribution carries the signature of how its benchmark was built. HotpotQA's answers are spans chosen by crowd workers and shipped without an alias list (Yang et al., 2018), so **<incomplete_answer_aliases>** reaches 15.9% of its samples. 2WikiMultiHopQA pairs a Wikidata triple with a Wikipedia summary and marks as gold the first sentence in which the object entity appears (Ho et al., 2020). It never checks whether the document states the relation in the triple, which leaves 15.7% of its samples with **<required_evidence_absent>**. MuSiQue draws its single-hop questions from other datasets (Trivedi et al., 2022), among them T-REx (Elsahar et al., 2018) and Zero Shot RE (Levy et al., 2017), which are themselves automatic alignments of Wikidata triples to Wikipedia text. MuSiQue's filter checks that the answer span appears in the paragraph, not that the paragraph states the relation, so **<required_evidence_absent>** reaches 39.7% of its samples. MuSiQue also composes several single-hop questions into one, which can drop the qualifiers that pinned the anchor entity down, so **<underspecified_question>** and **<ambiguous_question>** reach 5.4% and 5.8%, the highest of the three.

5.2 Statistics of the Released Benchmarks

Repaired benchmarks The *repaired* benchmarks apply the LLM Auditor's repairs to the full development sets and keep 22,255 of the 22,398 samples (99.4%), removing only the 143 for which the LLM Auditor produces no usable repair (Figure 6, left). A repair may change four fields, the question, the answer, the alias list and the gold marks. The documents themselves are never touched, so a repair never changes the text that a method retrieves from. Which field a repair changes follows the defect distribution of Section 5.1. HotpotQA's leading defect is a missing alias, so extending the alias list is also its most common repair, on 25% of its samples. The leading defect of 2Wiki-

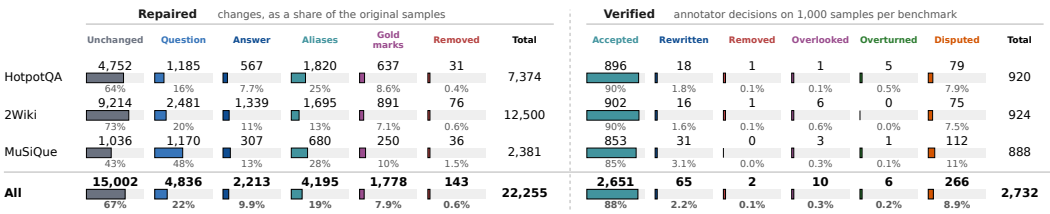


Figure 6: Changes made to the original benchmarks in the repaired and the verified versions. A repair often changes several fields, so a sample may appear in several columns of the left block.

MultiHopQA and MuSiQue, `<required_evidence_absent>`, cannot be repaired by adding documents, so the repair rewrites the question, and if needed the answer, to ask what the documents do support, on 20% and 48% of their samples respectively. The repaired benchmarks rest on the LLM Auditor alone, so a trained annotator outside the six of Section 4.2.2, who takes no part in refining the definitions, relabels the 300 random samples of Appendix A.5 under the twelve labels blind to its output. On whether a sample is defective, the LLM Auditor agrees with this blind annotation (Table 2 and Appendix D.4) at a Cohen's κ (Cohen, 1960) of 0.643, substantial agreement in the bands of Landis & Koch (1977). The repaired benchmarks are thus built on audits a human would largely agree with.

Verified benchmark All 3,000 samples reviewed in Section 4.2.2 are cross-checked by two annotators, and the verified benchmark keeps 2,732 of them (91.1%). The right block of Figure 6 reports what the annotators do with each reviewed sample. *Accepted* counts the samples on which both annotators keep the LLM Auditor's verdict and, if it proposes a repair, the verified benchmark keeps that repair unchanged. Appendix E.3 defines the others. The annotators accept 88% of the reviewed samples unchanged, so the great majority of the LLM Auditor's audits enter the release as proposed. The two annotators also agree closely with each other (Table 2 and Appendix E.4). They give the same clean-or-defective verdict on 94.4% of samples, with a Cohen's κ of 0.884, almost perfect agreement in the bands of Landis & Koch (1977). The verified benchmark keeps only samples on which their three-way judgments agree, so it gives a reliable basis for evaluating RAG methods.

Table 2: Agreement of the LLM Auditor with the blind annotation, and between two annotators.

Comparison	Samples	Raw agr.	Cohen's κ
LLM Auditor vs. blind annotation	300	0.830	0.643
Annotator vs. annotator	3,000	0.944	0.884

6 Re-evaluation on the Released Benchmarks

6.1 Methods and Setup

We re-evaluate nine state-of-the-art RAG methods on the original, repaired and verified benchmarks, three in each of three families: for retrieval structure, LinearRAG (Zhuang et al., 2026), GraphRAG (Edge et al., 2024) and HippoRAG 2 (Gutiérrez et al., 2025); for workflow, LogicRAG (Chen et al., 2026a), STRIDE (Chen et al., 2026b) and IRCoT (Trivedi et al., 2023); for agentic search, Search-R1 (Jin et al., 2025a), ReSearch (Chen et al., 2025) and Search-o1 (Li et al., 2025). Within each benchmark, every method retrieves from the same closed pool, which holds all of that benchmark's paragraphs with the gold marks stripped, at k=5 documents per query. Four methods share one BM25 index, and the rest use the retriever defined by their own method. Search-R1 and ReSearch train their own models, and we run their released checkpoints. The other seven run on Qwen3-32B, and one method per family, HippoRAG 2, IRCoT and Search-o1, also run on GPT-5.6. We also run four models under the same Oracle setting as in Section 2, the three models used there and Qwen3-32B, the backbone of most of these methods. We score the answers with an LLM judge (Ho et al., 2026), Llama-3.3-70B, from a different model family than any backbone here (Appendix F.3). Each method's implementation details are in Appendix F. Every GPT-5.6 call in this paper uses `gpt-5.6-sol`, at the reasoning effort given in Appendices A.4, A.5, D.1 and F.

Table 3: LLM-judge accuracy. *Full dev set* compares the original benchmark (orig.) with the repaired one (rep.); *1,000 samples* compares the 1,000 samples per benchmark drawn for human verification as published (orig.) and in the verified benchmark (ver.). Δ is the change from orig. Full results under all four metrics are in Appendix G.1.

		HotpotQA						**2WikiMultiHopQA**						**MuSiQue**					
		full dev set			1,000 samples			full dev set			1,000 samples			full dev set			1,000 samples		
Method	Base LLM	orig.	rep.	Δ	orig.	ver.	Δ	orig.	rep.	Δ	orig.	ver.	Δ	orig.	rep.	Δ	orig.	ver.	Δ
Retrieval structure																			
LinearRAG	Qwen3-32B	62.2	64.9	+2.7	62.3	66.5	+4.2	47.0	52.2	+5.2	47.6	52.3	+4.7	35.3	44.4	+9.1	34.7	46.6	+11.9
GraphRAG	Qwen3-32B	76.4	80.5	+4.1	76.5	81.6	+5.1	60.2	66.0	+5.8	59.0	65.8	+6.8	44.4	56.2	+11.8	44.3	57.2	+12.9
HippoRAG 2	Qwen3-32B	86.3	90.6	+4.3	85.4	91.0	+5.6	55.9	62.6	+6.7	57.2	65.0	+7.8	55.8	70.0	+14.2	55.5	70.5	+15.0
HippoRAG 2	GPT-5.6	93.2	97.3	+4.1	91.6	97.0	+5.4	93.0	96.7	+3.7	93.0	97.2	+4.2	74.1	85.9	+11.8	74.6	86.5	+11.9
Workflow																			
LogicRAG	Qwen3-32B	74.9	78.1	+3.2	73.1	77.3	+4.2	71.5	76.8	+5.3	70.5	77.1	+6.6	49.2	59.8	+10.6	50.3	62.0	+11.7
STRIDE	Qwen3-32B	75.8	79.4	+3.6	75.7	80.4	+4.7	69.4	75.1	+5.7	71.3	76.7	+5.4	52.0	61.8	+9.8	53.0	63.9	+10.9
IRCoT	Qwen3-32B	84.4	88.1	+3.7	83.7	87.9	+4.2	81.3	88.7	+7.4	81.2	88.2	+7.0	57.8	69.9	+12.1	57.8	70.8	+13.0
IRCoT	GPT-5.6	91.4	95.9	+4.5	90.3	95.3	+5.0	94.6	98.5	+3.9	94.6	98.7	+4.1	75.6	87.6	+12.0	77.1	88.9	+11.8
Agentic search																			
Search-R1	SearchR1-7B	69.1	71.7	+2.6	67.7	71.2	+3.5	30.5	32.5	+2.0	31.5	33.8	+2.3	31.7	41.5	+9.8	30.7	42.0	+11.3
ReSearch	ReSearch-32B	83.2	87.5	+4.3	82.0	88.3	+6.3	85.3	91.6	+6.3	84.5	91.0	+6.5	55.7	69.1	+13.4	54.3	69.0	+14.7
Search-o1	Qwen3-32B	79.8	83.9	+4.1	77.6	82.9	+5.3	81.7	88.0	+6.3	83.1	89.8	+6.7	52.2	62.9	+10.7	52.6	66.1	+13.5
Search-o1	GPT-5.6	89.3	93.3	+4.0	87.7	91.6	+3.9	92.1	95.9	+3.8	93.1	96.6	+3.5	66.8	80.3	+13.5	65.7	78.6	+12.9
Oracle Settings																			
Oracle	Qwen3-8B	91.3	95.0	+3.7	92.1	95.5	+3.4	82.3	91.9	+9.6	81.8	91.5	+9.7	69.0	84.6	+15.6	68.6	84.9	+16.3
Oracle	Qwen3-32B	92.4	96.6	+4.2	92.7	97.1	+4.4	87.3	97.5	+10.2	87.5	97.6	+10.1	70.7	86.5	+15.8	70.6	87.8	+17.2
Oracle	Qwen3-235B	93.4	96.9	+3.5	93.6	97.6	+4.0	88.7	97.6	+8.9	89.5	97.9	+8.4	76.5	90.0	+13.5	75.6	90.7	+15.1
Oracle	GPT-5.6	94.6	98.7	+4.1	95.2	99.0	+3.8	87.7	99.3	+11.6	88.9	99.5	+10.6	89.3	98.5	+9.2	89.9	98.4	+8.5

Table 4: LLM-judge accuracy on the verified benchmark as one benchmark. orig. scores the same samples in their original form, and Δ is the change from orig., with its 95% confidence interval. Unmarked RAG methods run on Qwen3-32B, and Search-R1 and ReSearch on their own checkpoints.

Retrieval structure				*Workflow*				*Agentic search*				*Oracle settings*			
Method	orig.	ver.	Δ	Method	orig.	ver.	Δ	Method	orig.	ver.	Δ	Base LLM	orig.	ver.	Δ
LinearRAG	49.0	55.2	+6.2 [5.0, 7.4]	LogicRAG	65.1	72.3	+7.2 [5.9, 8.4]	Search-R1	44.0	49.0	+5.0 [3.9, 6.3]	Qwen3-8B	81.7	90.7	+9.0 [7.8, 10.3]
GraphRAG	61.1	68.3	+7.2 [6.0, 8.5]	STRIDE	68.0	73.8	+5.8 [4.5, 7.0]	ReSearch	74.9	82.9	+8.0 [6.8, 9.3]	Qwen3-32B	84.3	94.3	+10.0 [8.8, 11.2]
HippoRAG 2	67.2	75.5	+8.3 [7.1, 9.6]	IRCoT	75.3	82.5	+7.2 [6.0, 8.4]	Search-o1	72.1	79.8	+7.7 [6.4, 9.0]	Qwen3-235B	86.9	95.5	+8.6 [7.4, 9.7]
HippoRAG 2 (GPT-5.6)	87.3	93.6	+6.3 [5.2, 7.4]	IRCoT (GPT-5.6)	88.3	94.4	+6.1 [5.1, 7.1]	Search-o1 (GPT-5.6)	82.9	89.1	+6.2 [5.1, 7.2]	GPT-5.6	91.4	99.0	+7.6 [6.6, 8.6]

6.2 Results before and after Repair

The original benchmarks understate every method. Every method in Table 3 scores higher on both the repaired and the verified benchmarks than on the originals, and the 95% confidence interval of every gain lies above zero (Appendix G.1). The gains do not depend on the samples the released benchmarks leave out, since they stay positive when the original benchmarks are scored only on the samples the released benchmarks keep (Appendix G.2). The understatement is largest where the defects are densest. For every RAG method the gain is largest on MuSiQue (57% defective).

Comparing RAG methods on the original benchmarks can be misleading. As shown in Table 3, the defects cost different methods different amounts, and such differences can decide which of two methods leads. On the original MuSiQue, IRCoT leads HippoRAG 2 by two points (57.8 against 55.8), but HippoRAG 2 gains 14.2 points from the repair and IRCoT 12.1, a difference of 2.1 points with a 95% confidence interval of 0.1 to 4.1 (Appendix G.1), so on the repaired benchmark the lead is gone (69.9 against 70.0). On the 1,000 samples behind the verified benchmark, IRCoT's lead of 2.3 points shrinks to 0.3. This case shows that a method trailing on the original benchmarks is not necessarily the weaker one, since it may simply have lost more points to the defects. Part of the blame placed on RAG systems therefore belongs to the benchmarks themselves.

The verified benchmark evaluates RAG methods against a valid standard. Drawn evenly from the three benchmarks, it can be read as one benchmark of 2,732 samples (Table 4), on which every method scores 5.0 to 10.0 points higher than on the original versions of the same samples. In the Oracle setting, GPT-5.6 answers 99.0% of these samples correctly, against 91.4% in their original form, so the benchmark itself now costs the methods almost nothing. The points that RAG methods still lose on the verified benchmark therefore reflect the methods rather than the benchmark.

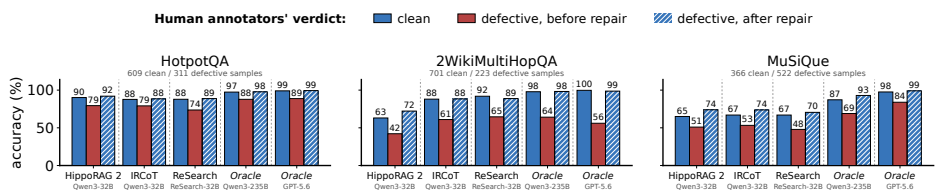


Figure 7: LLM-judge accuracy on the samples of the verified benchmark, grouped by the human annotators' verdict. The two defective bars cover the same samples before and after the repair.

6.3 The Repairs Fix the Defects

On the original benchmarks, the methods lose their points on the defective samples, and repairing these samples gives them back. We split the samples of the verified benchmark into a clean group and a defective group according to the human annotators' verdict (Figure 7), and run one method from each family in the RAG setting and two models in the Oracle setting. On the original benchmarks, the three RAG methods score 9 to 27 points lower on the defective group than on the clean group, and even with the gold documents in hand, GPT-5.6 answers only 56% of the defective samples of 2WikiMultiHopQA. The methods stay the same, and only the defective samples change. Once the defective group is repaired, the three RAG methods close most or all of the gap to the clean group, GPT-5.6 reaches 99% on the defective samples of 2WikiMultiHopQA, and Qwen3-235B, from a different model family than the LLM Auditor, scores at least as high on the repaired samples as on the clean ones. On the samples whose repair changes only the answer, the methods' original outputs are scored again, so the points recovered there come entirely from correcting the official answers (Appendix G.3). A large part of these lost points therefore comes from the defects themselves. Over the full development sets, the Oracle misses of Section 2 follow the same pattern, with 64.3% (Qwen3-8B) to 82.8% (GPT-5.6) of them on the 32.9% of samples that the LLM Auditor flags.

The recovery mainly comes from repairing defects that the samples already had, not from making the questions easier. Since the gains come mainly from the question-rewritten samples (Appendix G.3), we check whether their rewrites make the questions easier. The annotators record the reasoning hops of every repaired sample in the verified benchmark before and after its repair, and the great majority of repairs keep their number of hops (Table 21), while the rewrites almost never put the answer into the question (Appendix G.4). If the repairs made the questions easier, a model would answer them more easily even without the documents. Yet on all three repaired benchmarks, closed-book Qwen3-235B answers the rewritten questions slightly less accurately than the unchanged ones (Table 19). The gains therefore come mainly from repairing real defects, not from easier questions.

7 Conclusion

Retrieval-augmented generation (RAG) underpins agents that retrieve skills, memory and external services, yet HotpotQA, 2WikiMultiHopQA and MuSiQue, the multi-hop QA (question answering) benchmarks most widely used to evaluate RAG, contain many defects themselves. Each defective sample judges a RAG method against a wrong standard. The defects understate the methods, can change the outcome of comparisons between them and can send misleading signals into training. To address this, we propose a taxonomy of twelve defect labels and build on it a semi-automatic audit and repair framework. An LLM Auditor audits and repairs the full development sets of the three benchmarks, and human annotators then verify a subset of its work. Their cross-checking, with two annotators on every sample, gives the verified benchmark high and solid sample quality. Our re-evaluation shows that the repairs fix these defects. Our repaired and verified benchmarks let the research community evaluate RAG methods on data whose questions, evidence and answers hold together, so that comparisons between methods can be trusted. Our taxonomy and semi-automatic framework can also be used to audit other benchmarks at scale. Part of what the original benchmarks attribute to RAG failures reflects shortcomings in the benchmarks themselves.

AI USE STATEMENT

Generative AI is part of the method of this work. GPT-5.6 ran the exploratory analysis of the three development sets from which we distilled the twelve defect labels (Appendix A.5). As the LLM Auditor, GPT-5.6 also audited every benchmark record and generated the repairs, rewritten questions, answers and alias lists, on which the released benchmarks are built (Section 4.2). LLMs further serve as the backbones of the RAG methods we evaluate and as the LLM judge that scores their answers. We also used AI tools to polish the writing and revise the figures to make the paper easier to read, but all the ideas and designs in this work are our own. We manually checked and revised every part that was polished with AI. We did not use AI tools to build theoretical models or conceptual frameworks, to propose or refine hypotheses, to design the research methods or experiments, to search for related work, or to interpret the results. We take full responsibility for the final content of this work.

ETHICS STATEMENT

The three benchmarks we audited are publicly released research datasets built from Wikipedia articles. We release revised versions of them under the licenses of the source datasets, and we leave the supplied document text unchanged. Every released record keeps the identifier it has in the source benchmark, so it can be compared directly with the record it revises. All annotators who took part in this work were informed of the purpose of the project and of how their annotations would be used and released. They were trained on the twelve defect labels and their rules before their annotation began. The released records identify an annotator only by a pseudonymous identifier and carry no personal information.

REPRODUCIBILITY STATEMENT

We release the repaired and the verified benchmarks, the LLM Auditor's audit record for every sample, the human review records and the code of the LLM Auditor at `https://anonymous.4open.science/r/multihop-qa-benchmark_ICLR2027-4C2C`. The experimental details behind every number we report are given in the main text and in the appendices, in particular the LLM Auditor in Appendix D, the human verification protocol in Appendix E and the nine methods with their evaluation metrics in Appendix F.

REFERENCES

Sohyun An, Shuibenyang Yuan, Hayeon Lee, Cho-Jui Hsieh, and Alexander Min. Cycle-consistent search: Question reconstructability as a proxy reward for search agent training. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=IvfxffJKNp.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. Research: Learning to reason with search for llms via reinforcement learning. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/7b3a0d3a0864b66f229c0a9c84f9e950-Abstract-Conference.html.

Shengyuan Chen, Chuang Zhou, Zheng Yuan, Qinggang Zhang, Zeyang Cui, Hao Chen, Yilin Xiao, Jiannong Cao, and Xiao Huang. You don't need pre-built graphs for RAG: retrieval augmented generation with adaptive reasoning structures. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 30270–30278. AAAI Press, 2026a. doi: 10.1609/AAAI.V40I36.40278. URL https://doi.org/10.1609/aaai.v40i36.40278.

Wei Chen, Lili Zhao, Zhi Zheng, Huijun Hou, and Tong Xu. Stride: Strategic iterative decision-making for retrieval-augmented multi-hop question answering. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '26, pp. 202–212, New York, NY, USA, 2026b. Association for Computing Machinery. ISBN 9798400725999. doi: 10.1145/3805712.3809703. URL https://doi.org/10.1145/3805712.3809703.

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960. doi: 10.1177/001316446002000104. URL https://doi.org/10.1177/001316446002000104.

Yi Duan, Ying Liu, Zirui Tang, Haodong Chen, Jun Zhou, Yumou Liu, Bangrui Xu, Yukai Wu, Sidi Chen, Yuhan Zhou, et al. The last ai built by humans: Toward genuine recursive self-improvement. *arXiv preprint arXiv:2609.11873*, 2026.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. From local to global: A graph RAG approach to query-focused summarization. *CoRR*, abs/2404.16130, 2024. doi: 10.48550/ARXIV.2404.16130. URL https://doi.org/10.48550/arXiv.2404.16130.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. T-REx: A large scale alignment of natural language with knowledge base triples. In Nicoletta Calzolari, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga (eds.), *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://aclanthology.org/L18-1544/.

Tiantian Gan and Qiyao Sun. RAG-MCP: mitigating prompt bloat in LLM tool selection via retrieval-augmented generation. *CoRR*, abs/2505.03275, 2025. doi: 10.48550/ARXIV.2505.03275. URL https://doi.org/10.48550/arXiv.2505.03275.

Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous RL. In *First Workshop on Multi-Turn Interactions in Large Language Models*, 2025. URL https://openreview.net/forum?id=JSYCLJcniH.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *CoRR*, abs/2312.10997, 2023. doi: 10.48550/ARXIV.2312.10997. URL https://doi.org/10.48550/arXiv.2312.10997.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, Claire Barale, Robert McHardy, Joshua Harris, Jean Kaddour, Emile Van Krieken, and Pasquale Minervini. Are we done with MMLU? In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5069–5096, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.262. URL https://aclanthology.org/2025.naacl-long.262/.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=hkujvAPVsg.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. From RAG to memory: Non-parametric continual learning for large language models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267

of *Proceedings of Machine Learning Research*, pp. 21497–21515. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/gutierrez25a.html.

Kilem Li Gwet. Computing inter-rater reliability and its variance in the presence of high agreement. *British Journal of Mathematical and Statistical Psychology*, 61(1):29–48, 2008. doi: https://doi.org/10.1348/000711006X126600. URL https://bpspsychub.onlinelibrary.wiley.com/doi/abs/10.1348/000711006X126600.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In Donia Scott, Nuria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.580. URL https://aclanthology.org/2020.coling-main.580/.

Xanh Ho, Jiahao Huang, Florian Boudin, and Akiko Aizawa. Reassessing extractive QA datasets at scale: LLM-as-a-judge and in-depth analyses. In Simon Mille, Sebastian Gehrmann, Patrícia Schmidtová, Ondřej Dušek, Marzieh Fadaee, Kyle Lo, Enrico Santus, and Gabriel Stanovsky (eds.), *Proceedings of the Fifth Workshop on Generation, Evaluation and Metrics (GEM)*, pp. 84–101, San Diego, California, USA, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-423-1. doi: 10.18653/v1/2026.gem-main.9. URL https://aclanthology.org/2026.gem-main.9/.

Xinyi Hou, Yanjie Zhao, Shenao Wang, and Haoyu Wang. Model context protocol (mcp): Landscape, security threats, and future research directions. *ACM Trans. Softw. Eng. Methodol.*, 35(10), September 2026. ISSN 1049-331X. doi: 10.1145/3796519. URL https://doi.org/10.1145/3796519.

Yulong Hui, Chao Chen, Zhihang Fu, Yihao Liu, Jieping Ye, and Huanchen Zhang. Interact-RAG: Reason and interact with the corpus, beyond black-box retrieval. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=yHUjWb6eMe.

Abhinav Java, Srivathsan Koundinyan, Nagarajan Natarajan, and Amit Sharma. FrugalRAG: Less is more in RL finetuning for multi-hop question answering. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=uQKtwdJN0o.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Second Conference on Language Modeling*, 2025a. URL https://openreview.net/forum?id=Rwhi91ideu.

Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao Yang, and Ji-Rong Wen. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. In Guodong Long, Michale Blumestein, Yi Chang, Liane Lewin-Eytan, Zi Helen Huang, and Elad Yom-Tov (eds.), *Companion Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025 - 2 May 2025*, pp. 737–740. ACM, 2025b. doi: 10.1145/3701716.3715313. URL https://doi.org/10.1145/3701716.3715313.

Philipp Koehn. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu (eds.), *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 388–395, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-3250/.

Daeyong Kwon, Soyoung Yoon, and Seung won Hwang. Safe: An llm-as-verifier framework for evidence-grounded multi-hop reasoning, 2026. URL https://arxiv.org/abs/2604.01993.

J. Richard Landis and Gary G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977. ISSN 0006341X, 15410420. URL http://www.jstor.org/stable/2529310.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In Roger Levy and Lucia Specia (eds.), *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pp. 333–342, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/K17-1034. URL https://aclanthology.org/K17-1034/.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 5420–5438, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.276. URL https://aclanthology.org/2025.emnlp-main.276/.

Md Mahadi Hasan Nahid and Davood Rafiei. Prism: Agentic retrieval with llms for multi-hop question answering, 2026. URL https://arxiv.org/abs/2510.14278.

Curtis G Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. URL https://openreview.net/forum?id=XccDXrDNLek.

Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems. *CoRR*, abs/2310.08560, 2023. doi: 10.48550/ARXIV.2310.08560. URL https://doi.org/10.48550/arXiv.2310.08560.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=dHng2O0Jjr.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475. URL https://aclanthology.org/2022.tacl-1.31/.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10014–10037, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.557. URL https://aclanthology.org/2023.acl-long.557/.

Venktesh V, Deepali Prabhu, and Avishek Anand. DEXTER: A benchmark for open-domain complex question answering using llms. *CoRR*, abs/2406.17158, 2024. doi: 10.48550/ARXIV.2406.17158. URL https://doi.org/10.48550/arXiv.2406.17158.

Joshua Vendrow, Edward Vendrow, Sara Beery, and Aleksander Madry. Do large language model benchmarks test reliability? *CoRR*, abs/2502.03461, 2025. doi: 10.48550/ARXIV.2502.03461. URL https://doi.org/10.48550/arXiv.2502.03461.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=ehfRiF0R3a.

Liang Wang, Haonan Chen, Nan Yang, Xiaolong Huang, Zhicheng Dou, and Furu Wei. Chain-of-retrieval augmented generation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=gUPGGCM4WH.

Chuanjie Wu, Zhishang Xiang, Yunbo Tang, Zerui Chen, Qinggang Zhang, and Jinsong Su. Memgraphrag: Memory-based multi-agent system for graph retrieval-augmented generation. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '26, pp. 5407–5418, New York, NY, USA, 2026a. Association for Computing Machinery. ISBN 9798400722592. doi: 10.1145/3770855.3818074. URL https://doi.org/10.1145/3770855.3818074.

Peilin Wu, Mian Zhang, Kun Wan, Wentian Zhao, Kaiyu He, Xinya Du, and Zhiyu Chen. HiPRAG: Hierarchical process rewards for efficient agentic retrieval augmented generation. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=Gt4v9WBPzm.

Yutao Xie, Nathaniel Thomas, Nicklas Hansen, Yang Fu, Li Erran Li, and Xiaolong Wang. TIPS: Turn-level information-potential reward shaping for search-augmented LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=eBMOr6a84z.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

Fengji Zhang, Xinyao Niu, Chengyang Ying, Guancheng Lin, Zhongkai Hao, Zhou Fan, Chengen Huang, Jacky Keung, Bei Chen, and Junyang Lin. A^{2}search: Ambiguity-aware question answering with reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=3CPzUWIoNf.

Liangliang Zhang, Zhuorui Jiang, Hongliang Chi, Haoyang Chen, Mohammed Elkoumy, Fali Wang, Qiong Wu, Zhengyi Zhou, Shirui Pan, Suhang Wang, and Yao Ma. Diagnosing and addressing pitfalls in KG-RAG datasets: Toward more reliable benchmarking. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025a. URL http://papers.nips.cc/paper_files/paper/2025/hash/61a57ab030670598c4200b31100d254c-Abstract-Datasets_and_Benchmarks_Track.html.

Nan Zhang, Prafulla Kumar Choubey, Alexander Fabbri, Gabriel Bernadett-Shapiro, Rui Zhang, Prasenjit Mitra, Caiming Xiong, and Chien-Sheng Wu. SireRAG: Indexing similar and related information for multihop reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=yp95goUAT1.

Shaochen (Henry) Zhong, Yifan Lu, Lize Shao, Bhargav Bhushanam, Xiaocong Du, Yixin Wan, Yucheng Shi, Daochen Zha, Yiwei Wang, Ninghao Liu, Kaixiong Zhou, Shuai Xu, Kai-Wei Chang, Louis Feng, Vipin Chaudhary, and Xia Hu. Mquake-remastered: Multi-hop knowledge editing can only be advanced with reliable evaluations. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=m9wG6ai2Xk.

Luyao Zhuang, Shengyuan Chen, Yilin Xiao, Huachi Zhou, Yujing Zhang, Hao Chen, Qinggang Zhang, and Xiao Huang. LinearRAG: Linear graph retrieval augmented generation on large-scale corpora. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=mCtfkypdm6.

APPENDIX CONTENTS

A The Three Benchmarks and the Evidence of Their Defects **17**
A.1 Benchmark Usage and Sample Structure . . . 17
A.2 Provenance and Statistics . . . 17
A.3 The Full HotpotQA Record in Figure 1 . . . 18
A.4 Oracle and Closed-Book Results in Full . . . 18
A.5 The Origin of the Labels and the 300-Sample Hand Audit . . . 19

B Extended Related Work **19**

C The Twelve Defect Labels **20**
C.1 Query Family . . . 21
C.2 Hop Family . . . 23
C.3 Pointer Family . . . 27
C.4 Answer Family . . . 28

D The LLM Auditor **31**
D.1 The Three Calls and Their Setup . . . 31
D.2 Prompts . . . 31
D.3 Automatic Checks . . . 32
D.4 Agreement with the Blind Annotation . . . 32

E The Human Verification Protocol **33**
E.1 Sampling and Batch Construction . . . 33
E.2 Sample Sizes in Prior Work . . . 33
E.3 Reviewing One Sample . . . 33
E.4 Agreement between Annotators . . . 34

F Re-evaluation Setup **35**
F.1 The Nine Methods . . . 35
F.2 Implementation . . . 35
F.3 Evaluation Metrics . . . 37

G Re-evaluation Results in Full **37**
G.1 Full Results on the Two Released Benchmarks under Four Metrics . . . 37
G.2 Gains on Samples Matched between the Original and Released Benchmarks . . . 38
G.3 Where the Gains Come From . . . 39
G.4 Do the Repairs Make the Questions Easier? . . . 41

A THE THREE BENCHMARKS AND THE EVIDENCE OF THEIR DEFECTS

A.1 BENCHMARK USAGE AND SAMPLE STRUCTURE

HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020) and MuSiQue (Trivedi et al., 2022) are the benchmarks on which multi-hop RAG methods are developed and compared. Each sample in the three benchmarks is built around the same three parts, a **query** (the question text), a set of **documents** (Wikipedia paragraphs split into gold documents that carry the evidence and distractor documents that do not), and an **official answer**. A HotpotQA or 2WikiMultiHopQA sample includes ten paragraphs and a MuSiQue sample twenty in nearly every case, of which exactly two are gold in HotpotQA, two or four in 2WikiMultiHopQA, and two to four in MuSiQue. We took HotpotQA from its *distractor* release, which supplies each sample with its own paragraphs, rather than from its *fullwiki* release (Yang et al., 2018), and MuSiQue in its *answerable* variant, which excludes the deliberately unanswerable questions of MuSiQue-Full (Trivedi et al., 2022). For evaluation, the paragraphs of all samples were pooled into one retrieval corpus per benchmark, as Section 6.1 describes, and Table 6 gives the size of each pool.

Beyond these three parts, the benchmarks differ in what else they include. An **official reasoning chain**, recording the intended step-by-step path from the question to the answer, comes with every 2WikiMultiHopQA and MuSiQue sample and with no HotpotQA sample. A list of **answer aliases**, the alternative surface forms that a scorer also accepts, is supplied for most 2WikiMultiHopQA samples and for a minority of MuSiQue samples, while no HotpotQA sample carries one; Appendix A.2 gives the exact shares.

The three benchmarks also mark the supporting evidence inside a gold document at different granularities. HotpotQA and 2WikiMultiHopQA mark individual sentences, so one gold mark selects one sentence of one paragraph, whereas MuSiQue marks the gold paragraph as a whole. These marks form an annotation layer that sits on top of the document text and can be wrong on its own, which is where the Pointer-family defects of Section 4.1 arise.

A.2 PROVENANCE AND STATISTICS

Provenance and preprocessing Each of the three papers links to a project page or repository rather than to the data files, so we followed each link to the source it points at. HotpotQA is the exception. Its project page is still online, but the host it links to for the data files no longer responds, so we took the distractor development set from the Hugging Face distribution of the benchmark and rebuilt it into the official native JSON schema. Table 5 records the archives and mirror used by the downloader in our release.

The 2WikiMultiHopQA archives are available from its official repository,[1] MuSiQue from the Google Drive archive that its own download script fetches,[2] and the HotpotQA distribution we used from the Hugging Face Hub.[3] The December 2020 2WikiMultiHopQA archive we audited is also the one behind FlashRAG's release of the benchmark (Jin et al., 2025b).[4] We compared the two record by record and found them identical on all 12,576 development samples, in the titles and the order of the paragraphs, in the segmentation of their sentences and in the supporting-fact pointers. FlashRAG is the evaluation environment that several recent methods build on, among them Search-R1 (Jin et al., 2025a) and ReSearch (Chen et al., 2025), so the records we audited are the records those methods are scored on. The three development sets were then converted into one common format, so that the audit read a single schema. That conversion only renamed and rearranged fields, so the defects passed through unchanged. Every defect reported in this paper is therefore a property of the published benchmark rather than of our preprocessing.

Statistics Table 6 summarizes the three development sets, which hold 22,398 samples between them. For each benchmark it gives the number of unique documents in the pooled corpus, the

[1] https://github.com/Alab-NII/2wikimultihop

[2] https://github.com/StonyBrookNLP/musique

[3] https://huggingface.co/datasets/hotpotqa/hotpot_qa

[4] https://huggingface.co/datasets/RUC-NLPIR/FlashRAG_datasets

Table 5: Input snapshots used for the audit.

Benchmark	Input snapshot
HotpotQA	Distractor configuration, validation split, from the Hugging Face dataset `hotpotqa/hotpot_qa`; 7,405 samples, rebuilt into the official native JSON schema.
2WikiMultiHopQA	December 2020 `data_ids.zip`: `dev.json` and `id_aliases.json`.
MuSiQue	Answerable development set, `musique_ans_v1.0_dev.jsonl`, from the official `musique_v1.0.zip` archive.

share of samples that carry answer aliases, the distribution of official chain lengths, and the official question types.

Table 6: Statistics of the three original benchmarks.

	HotpotQA	2WikiMultiHopQA	MuSiQue
Samples	7,405	12,576	2,417
Documents in the corpus	66,635	56,707	21,100
Answer aliases	none	84% of samples	28% of samples
Official chain	none	2 hops 76.3%, 4 hops 22.3%, other 1.4%	2 to 4 hops, given by the type name
Question types	`bridge` 79.9% `comparison` 20.1%	`compositional` 41.6% `comparison` 24.2% `bridge_comparison` 21.9% `inference` 12.3%	`2hop` 51.8% `3hop1` 23.5% `4hop1` 10.2% `3hop2` 7.9% `4hop3` 3.9% `4hop2` 2.6%

A.3 The Full HotpotQA Record in Figure 1

The defective sample `ID: 5a7269fa5542992359bc308d` **(HotpotQA)**

Query: Which tennis player was born first, Kimberly Po or Leoš Friedl?

Gold documents:

⋆ [Leoš Friedl] "Leoš Friedl (born 1 January 1977 in Jindřichův Hradec) is an inactive Czech professional tennis player best known for his doubles play with František Čermák."

⋆ [Kimberly Po] "Kimberly Po-Messerli (born October 20, 1971) is a former professional tennis player from the United States."

Answer supported by the documents: Kimberly Po

Official answer: Leoš Friedl (aliases: none)

Where it breaks: the official answer. Both birth dates are stated, and comparing them gives one answer: Po was born in 1971 and Friedl in 1977, so Po was born first. The official answer names Friedl instead.

A.4 Oracle and Closed-Book Results in Full

Figure 2 plots Cover-EM only, and Table 7 reports all four metrics for all three backbones under both conditions. Each Oracle prompt contains the question and the complete text of that sample's gold documents, with the distractor documents removed and the official answer withheld; the closed-book prompt contains the question alone.

The two Qwen3 models were served locally at temperature zero with thinking mode disabled; GPT-5.6 (`gpt-5.6-sol`) was called through its hosted API at reasoning effort `xhigh`.

Table 7: Oracle vs. closed-book results on all 22,398 samples, for three backbones under all four metrics, where Judge is the LLM judge.

	HotpotQA				2WikiMultiHopQA				MuSiQue				All			
Model	EM	Cover-EM	F1	Judge	EM	Cover-EM	F1	Judge	EM	Cover-EM	F1	Judge	EM	Cover-EM	F1	Judge
Oracle: the question and its gold documents																
Qwen3-8B	54.0	73.0	71.6	91.3	64.4	79.9	74.6	82.3	37.3	57.4	50.4	69.0	58.1	75.2	71.0	83.8
Qwen3-235B	62.2	74.3	78.2	93.4	75.4	86.4	83.0	88.7	51.7	66.0	63.5	76.5	68.5	80.2	79.3	89.0
GPT-5.6	52.1	80.5	73.4	94.6	62.7	86.5	74.1	87.7	54.8	82.1	73.3	89.3	58.3	84.1	73.8	90.1
Closed-book: the question alone																
Qwen3-8B	14.6	23.7	24.8	34.8	19.9	29.4	27.8	30.0	4.4	7.7	12.1	17.6	16.5	25.2	25.1	30.3
Qwen3-235B	28.7	36.8	40.8	53.5	35.8	41.8	43.8	44.3	10.4	16.5	20.6	32.0	30.7	37.4	40.3	46.0
GPT-5.6	36.4	61.8	58.3	83.3	50.4	84.4	70.2	87.9	23.7	46.3	42.6	62.2	42.9	72.8	63.3	83.6

Prior Oracle studies Two prior studies run the same Oracle setting and report the same shortfall. **PRISM** (Nahid & Rafiei, 2026) runs it with GPT-4o on 500 samples per benchmark, scored by EM and F1 only, and reports it as an "*oracle-context*" condition in which the model is "*provided only with the gold supporting evidence required to answer the question*". It treats this condition as the target that its retrieval narrows the gap to, and attributes the high error rates on HotpotQA and MuSiQue that remain once all gold evidence is retrieved to "*the greater reasoning complexity of these datasets*". **DEXTER** (V et al., 2024) feeds gold contexts to gpt-3.5-turbo on MuSiQue-Ans (1,252 samples) and 2WikiMultiHopQA (1,200 two-hop samples), scored by Cover-EM, and sets up its Oracle run "*to obtain an upper bound on LLM reasoning performance*". Neither work asks whether the errors that remain come from the samples themselves. Section 2 argues that they largely do.

A.5 The Origin of the Labels and the 300-Sample Hand Audit

We first distilled the twelve labels of Section 4.1 from an exploratory analysis of the full development sets of the three benchmarks, which GPT-5.6 ran with the same configuration as the LLM Auditor (Appendix D). We then reviewed the labels and definitions this analysis proposed. To refine them, we randomly drew 100 samples from the development set of each benchmark and audited all 300 by hand. For each sample we read in full the question, every supplied document (gold and distractor alike), the official reasoning chain where one exists and the official answer with its aliases, and checked every problem we found against the proposed labels to sharpen their definitions and the boundaries between them. Once the definitions were fixed, a trained annotator outside the six of Section 4.2.2, who took no part in refining them, relabeled the same 300 samples under the twelve labels without seeing the output of the LLM Auditor. A sample is *defective* if at least one label applies, and *clean* otherwise. Under this relabeling, a large share of every benchmark is defective (Figure 8), nearly two thirds of MuSiQue (63%), about a third of HotpotQA (35%) and a quarter of 2WikiMultiHopQA (24%).

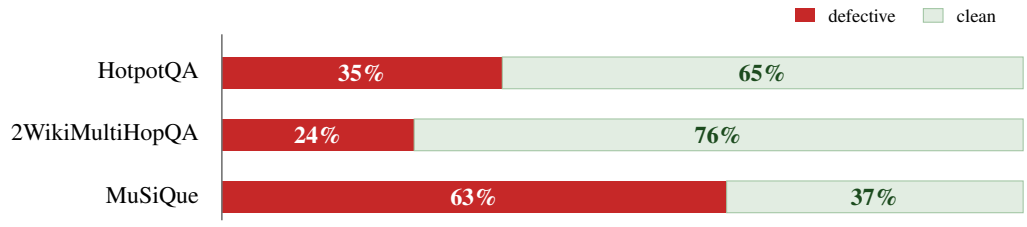


Figure 8: Share of clean vs. defective samples among the 100 samples per benchmark, relabeled by a trained annotator under the definitions of the twelve defect labels.

B Extended Related Work

This appendix discusses the works of Section 3 in more detail.

Audits of the multi-hop QA benchmarks Platinum (Vendrow et al., 2025) audits a 250-question sample of HotpotQA, removing 66 bad questions and keeping the other 184, 6 of them with corrected labels. How Platinum chooses what to inspect matters as much as the count. A question is inspected only when at least one frontier model misses it, so a defective question on which every model happens to match the official answer is never inspected, and the procedure can only put a lower bound on how much of the benchmark is defective. SAFE (Kwon et al., 2026) trains a verifier that checks a model's reasoning step by step during multi-hop QA, rather than judging only the final answer afterward. Training that verifier needs clean supervision, so SAFE first runs an LLM pipeline over the three benchmarks, testing whether a chain of triples from the gold documents leads from the question to the answer. It states that "*3%–14% of instances are unverifiable under our atomic grounding constraints*", and its rates on the validation splits are 3.09% on HotpotQA, 7.16% on 2WikiMultiHopQA and 14.18% on MuSiQue, so the peak sits on the same benchmark where our own audit found the highest defect rate. SAFE reaches those figures automatically and reports no human validation of them. It drops the flagged instances from its own training and evaluation data while releasing "*the original dataset with our feedback labels as an augmented resource*". We start from the same observation but make the defects our subject, identifying and repairing them in the original benchmarks.

Audits in other domains Northcutt et al. (2021) estimate that at least 3.3% of labels are wrong on average across ten test sets and at least 6% on the ImageNet validation set. Their validation is crowdsourced, five workers per flagged example with a three-of-five threshold, and no agreement coefficient is reported; the only inter-rater agreement figure the paper gives is that experts agreed on 77% of the ImageNet images they reviewed. MMLU-Redux (Gema et al., 2025) estimates that 6.49% of MMLU questions are erroneous, with the rate reaching 57% in the Virology subset alone, and it is the only one of the audits discussed here to report a chance-corrected agreement coefficient for its human labels, Cohen's κ among three annotators on five subjects. KG-RAG Pitfalls (Zhang et al., 2025a) audits over a thousand items drawn from sixteen KGQA benchmarks, each item reviewed by two annotators with disagreements resolved by discussion, finds average factual correctness of 57%, as low as 52% on WebQSP and 49% on CWQ, and answers the problem by generating a replacement benchmark rather than repairing the originals.

MQuAKE-Remastered (Zhong et al., 2025) reports that "*up to 33% or 76% of MQuAKE's questions and ground truth labels are, in fact, corrupted in various fashions due to some unintentional clerical or procedural oversights*". It names five error types: contamination between edited and unedited cases, contamination between different edited cases, conflicting edits, missing information in the question and duplicated cases. The two kinds of contamination arise only when several cases are edited together, so they are properties of the evaluation protocol rather than of a published record, and they can be computed from the dataset's own edit bookkeeping; both headline figures come from them, 76% being the share of MQuAKE-T cases contaminated when 100 cases are edited and 33% the share of MQuAKE-CF-3k cases contaminated when all 3,000 are edited. The paper corrects conflicting edits and duplicated cases by hand, stating that they affect fewer than fifty cases per type per dataset, and restores the missing information by rewriting every question with an LLM and inspecting by hand the few rewrites its checks still flag. Re-benchmarking two methods on both versions, the paper finds that both score higher on the repaired data, with the differences "*especially pronounced for stronger methods*", and concludes that the original benchmark "*cannot faithfully reflect the true capacities of the methods*". We find the same understatement on passage-grounded RAG, which suggests it is not particular to knowledge editing. The defects themselves do not carry over. MQuAKE supplies no documents, so the six labels of our Hop and Pointer families, half the taxonomy, have no counterpart there.

C THE TWELVE DEFECT LABELS

This appendix gives the full definitions of the twelve defect labels introduced in Section 4.1. For each label we state the definition and the boundary rules that separate it from neighboring labels, show in a schematic where it breaks the chain and give a worked example from one of the benchmarks. As in Section 4.1, we read a two-hop sample as one chain running from the question to the answer (Figure 9). In a healthy chain, the question names the anchor A and the relations r_1 and r_2, the documents supply the bridge B and the answer entity C, and the official answer records C.

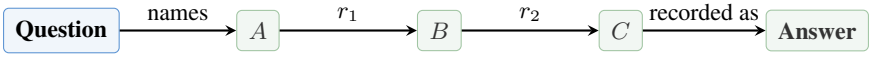


Figure 9: The healthy two-hop chain that every schematic in this appendix starts from.

C.1 Query Family

Query-family defects leave the question without a single chain to follow, whether the text is too broken to read, the anchor A matches more than one entity in the documents or the relation r yields more than one supported value. Unlike in the Hop family, the documents may contain every fact the chain needs, and the problem is that the question does not identify one chain among them.

C.1.1 MALFORMED_QUESTION

The question text itself is broken badly enough to change or obscure what is being asked, for example truncated, ungrammatical, with a modifier attached to the wrong phrase or carrying a critical typo (Figure 10). This is checked on the wording alone, before the anchor A or the relations are examined; awkward but recoverable wording is not a defect.

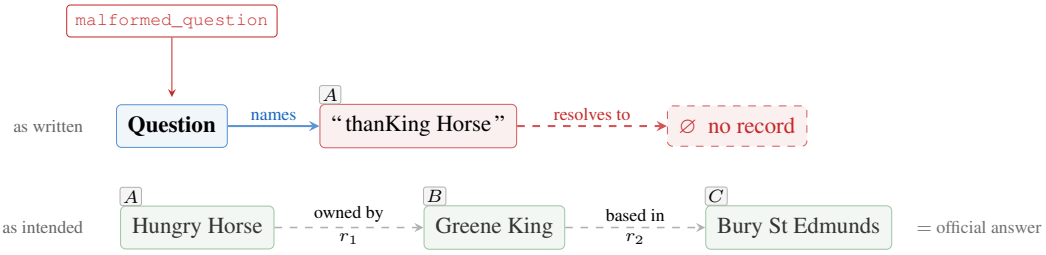


Figure 10: `malformed_question`. Followed as written, the question reaches no entity, while the reasoning chain it intends stays intact.

`malformed_question` `ID: 5ae24abc5542994d89d5b3c3` **(HotpotQA)**

Query: Where is the company that owns thanKing Horse pub restaurant chain based ?

Gold documents:

⋆ [Hungry Horse] “Hungry Horse is a chain of 225 pub-restaurants in England, Wales and Scotland, owned by Greene King Brewery. It was founded in 1995, and promotes itself as offering low cost meals for families and groups.”

⋆ [Greene King] “Greene King is the UK’s largest pub retailer and brewer. It is based in Bury St Edmunds, Suffolk, England. The company owns pubs, restaurants and hotels.”

Official answer: Bury St Edmunds

Where it breaks: the restaurant chain the question asks about is written “thanKing Horse”, a misspelling visible from the wording alone. The documents confirm that it names nothing at all, since they describe a restaurant chain called Hungry Horse and no document connects the two names. Every other part of the sample is healthy, and the two hops run cleanly from Hungry Horse to Greene King to Bury St Edmunds. Only the question text is broken.

C.1.2 UNDERSPECIFIED_QUESTION

A name or description in the question, even with all its qualifiers, matches two or more entities in the supplied documents, so the anchor A cannot be pinned to a unique referent (Figure 11). The chain therefore forks at its very start, on the anchor, before any relation is followed. In the worked example, the film title in the question belongs to two different films.

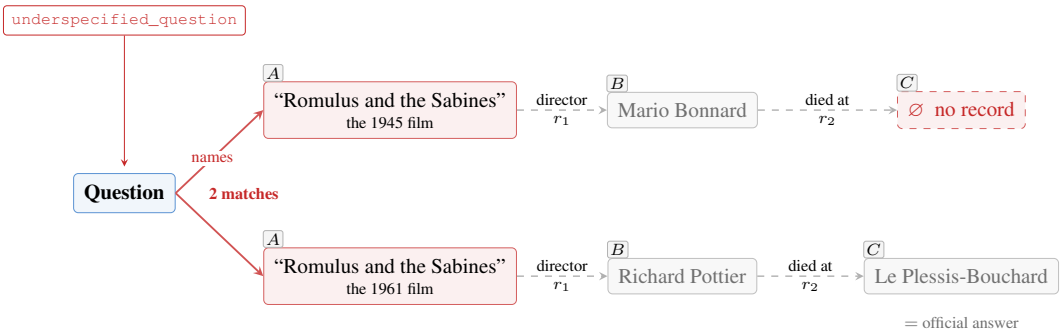


Figure 11: `underspecified_question`. The title in the question matches two films, so the chain forks at the anchor before any relation is followed.

`underspecified_question` `ID: 16915f740bdc11eba7f7acde48001122` **(2WikiMultiHopQA)**

Query: Where was the place of death of the director of film Romulus And The Sabines?

Documents: two films carry that title (⋆ gold, ○ not gold).

○ [Romulus and the Sabines (1945 film)] "Romulus and the Sabines (Italian: Il ratto delle sabine) is a 1945 Italian comedy film directed by Mario Bonnard and starring Totò, Carlo Campanini and Clelia Matania."

⋆ [Romulus and the Sabines] "Il ratto delle sabine is an Italian adventure comedy film from 1961, directed by Richard Pottier, written by Edoardo Anton, starring Roger Moore and Jean Marais."

⋆ [Richard Pottier] "Richard Pottier (6 June 1906, Graz – 2 November 1994, Le Plessis-Bouchard) was an Austrian-born French film director."

Official reasoning chain:

hop 1: Romulus and the Sabines → director → *Richard Pottier*

hop 2: Richard Pottier → place of death → *Le Plessis-Bouchard*

Official answer: Le Plessis-Bouchard

Where it breaks: the question names the film as "Romulus And The Sabines" and stops there, with no year, no director and no other qualifier. The documents describe two films with that title, one from 1945 and one from 1961, each with its own director. The anchor of hop 1 therefore matches two entities, and the relation is not at fault, since each film has exactly one director. The official chain silently takes the 1961 film, which works only because the documents happen to give a place of death for Pottier and not for Bonnard.

C.1.3 AMBIGUOUS_QUESTION

Every mention in the question resolves to a unique anchor A, but following the relation yields two or more supported values, either because the relation has several values in the documents or because the relation word allows several readings (Figure 12). This hop therefore yields no unique value. The fork sits one step later than in `underspecified_question`, on the relation r, and the position of the fork is the whole difference between the two labels. A question about a title that matches two different films forks at the anchor and is labeled `underspecified_question`, while a question about a unique film with two directors forks at the relation and is labeled `ambiguous_question`.

`ambiguous_question` `ID: 2hop__307719_121900` **(MuSiQue)**

Query: What college did the 2011 Australian Open - Men's Doubles winner go to?

Gold documents:

⋆ [2011 Australian Open – Men's Doubles] "Bob and Mike Bryan, the two-time defending champions in this event, successfully defended their title. They defeated Mahesh Bhupathi and Leander Paes 6–3, 6–4 in the final."

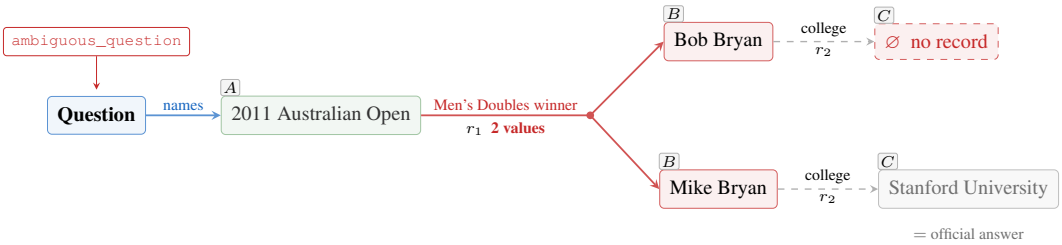


Figure 12: `ambiguous_question`. The anchor is unique, but the winner relation yields two supported values.

⋆ [Mike Bryan] "Mike Bryan played for Stanford University in 1997 and 1998, where he helped the Cardinal win back-to-back NCAA team championships. In 1998, he won the NCAA doubles title with his twin brother Bob."

Official reasoning chain:

hop 1: 2011 Australian Open – Men's Doubles → winner → *Mike Bryan*

hop 2: Mike Bryan → college → *Stanford University*

Official answer: Stanford University

Where it breaks: the anchor is fine, since the 2011 Australian Open men's doubles is one specific event. The fork is at the relation instead, because a doubles title is won by two players and the documents name both of them, Bob and Mike Bryan. The official chain takes Mike Bryan and drops Bob without saying why, which works only because no document gives Bob's college.

C.2 Hop Family

In Hop-family defects, the chain cannot be completed inside the documents, either because one hop fails or because the two hops do not join. In the gold documents, the chain of Figure 9 is realized as two facts, normally stated in two different documents:

$$\text{document 1: } A \xrightarrow{r_1} B \qquad \text{document 2: } B' \xrightarrow{r_2} C$$

In a healthy sample the value B produced by hop 1 and the anchor B' used by hop 2 are the same entity, $B = B'$, so the two hops are joined by a bridge and the two gold documents chain into one path from the question to the answer.

We read that bridge with a fixed rule, which we call the bridge convention. When the value produced by hop 1 and the anchor used by hop 2 carry the same name, they are taken to be the same entity, and no sentence has to say so. Multi-hop samples rarely contain a sentence whose only job is to identify the two, and the absence of such a sentence is normal rather than a defect. A defect is found only when the documents show that the two are not the same.

C.2.1 INSUFFICIENT_QUESTION_EVIDENCE_LINKAGE

Each hop is stated in its own document, but the documents themselves show that the value B of hop 1 and the anchor B' of hop 2 are two different entities that merely share a name, so the two hops never join into one chain (Figure 13). The evidence is rarely a sentence stating that the two are not the same, since documents seldom contain one; it is usually a detail showing that the two cannot be the same thing, most often a mismatch of profession, place or date. Where the documents carry no such evidence, the bridge convention of the Hop family stands and this label does not apply.

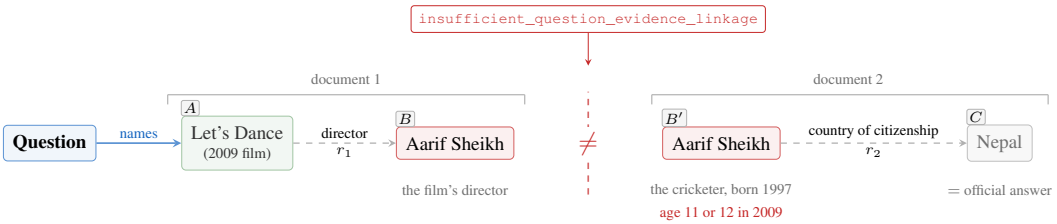


Figure 13: `insufficient_question_evidence_linkage`. Each hop holds on its own, but the film's director and the cricketer are two different people who share a name, so the bridge between the hops fails.

`insufficient_question_evidence_linkage` `ID: 205e0f460bde11eba7f7acde48001122` **(2WikiMultiHopQA)**

Query: What nationality is the director of film Let'S Dance (2009 Film)?

Gold documents:

⋆ [Let's Dance (2009 film)] "Let's Dance is a 2009 Bollywood film that stars Gayatri Patel, Ajay Chaudhary and Aquib Afzal in the lead roles. It is directed and edited by Aarif Sheikh."

⋆ [Aarif Sheikh] "Aarif Sheikh (born 5 October 1997) is a Nepalese cricketer, who was the vice-captain of the Nepal national under-19 cricket team, and currently plays for the senior team. All-rounder Aarif is a right-handed batsman and a right-arm medium pace bowler. He made his debut for Nepal against Hong Kong in May 2014."

Official reasoning chain:

hop 1: Let's Dance (2009 film) → director → *Aarif Sheikh*

hop 2: Aarif Sheikh → country of citizenship → *Nepal*

Official answer: Nepal

Where it breaks: both hops are individually fine, since document 1 states who directed the film and document 2 states a nationality. Read together, however, the documents show that the film's director and the cricketer are different people who share the same name. The film was released in 2009, and the cricketer, born in October 1997, would have been 11 or 12 years old, an implausible age for its director. The official chain still treats them as one person, so its answer, Nepal, is the cricketer's nationality, and the documents never give the director's.

C.2.2 REQUIRED_EVIDENCE_ABSENT

The anchor of a hop is fixed, either A read from the question or B correctly produced by hop 1, and the relation is clear, but no supplied document states the fact at the far end, so following the relation yields no supported value (Figure 14):

$$A \xrightarrow{r_1} \varnothing \qquad \text{or} \qquad B \xrightarrow{r_2} \varnothing$$

Two rules apply. First, if the documents state only a different relation, even a close one, and never the relation the question asks about, the label still applies. Second, the search covers every supplied document, not only the gold-marked ones; if the fact appears in text that carries no gold mark, the defect belongs to the Pointer family instead.

`required_evidence_absent` `ID: bd9609880bdb11eba7f7acde48001122` **(2WikiMultiHopQA)**

Query: Who is the father of the director of film Battle In Seattle?

Gold documents:

⋆ [Battle in Seattle] "Battle in Seattle is a 2007 political action-thriller film written and directed by Stuart Townsend, in his directorial debut. The story is based on the protest activity at the WTO Ministerial Conference of 1999."

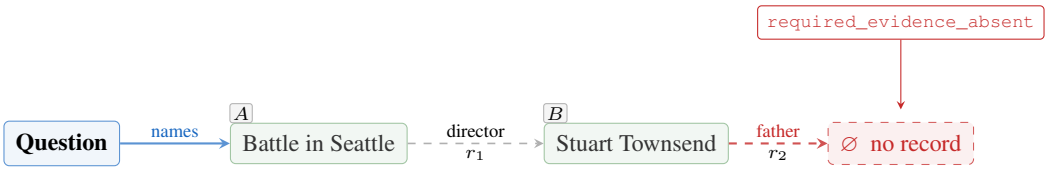


Figure 14: `required_evidence_absent`. Hop 1 is supported, but no supplied document identifies Stuart Townsend's father, leaving hop 2 unsupported.

⋆ [Stuart Townsend] "Stuart Peter Townsend (born 15 December 1972) is an Irish actor. His most notable portrayals are of the characters Lestat de Lioncourt in the 2002 film adaptation of Anne Rice's 'Queen of the Damned', and Dorian Gray in the 2003 film adaptation of Alan Moore's 'The League of Extraordinary Gentlemen'."

Six of the eight non-gold documents state a *father* relation, always about someone else:

∘ [Cleomenes II] "He was the father of Acrotatus I, the father of Areus I, and of Cleonymus, the father of Leonidas II." ∘ [Paul Brooke] "He is the father of actor Tom Brooke." ∘ [Peter Hamel] "He is the father of the composer Peter Michael Hamel." ∘ [Lars Eliasson] "He is the father of the later Member of Parliament Anna Eliasson." ∘ [Obata Toramori] "He was the father of Obata Masamori." ∘ [Yasuichi Oshima] "He is the father of manga artist Towa Oshima."

Official reasoning chain:

hop 1: Battle in Seattle → director → *Stuart Townsend*

hop 2: Stuart Townsend → father → *Peter Townsend*

Official answer: Peter Townsend (aliases: Peter Townsend (golfer))

Where it breaks: hop 1 is stated and hop 2 is not. The film document names Stuart Townsend as the director, but no supplied document says who his father is. The official answer, Peter Townsend, therefore rests on a fact that the documents never give.

C.2.3 EVIDENCE_CONTRADICTION

The anchor and the relation of a hop are both fixed, and the relation allows only one value by nature, but the documents give two or more values that cannot all be true (Figure 15). A person, for instance, dies only once. If the relation can legitimately hold several values, as a film can have two directors, the values do not conflict and the label is `ambiguous_question`.

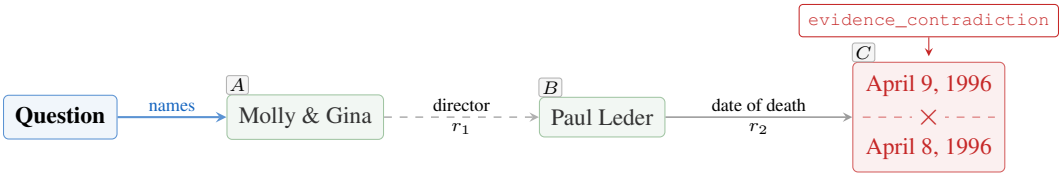


Figure 15: `evidence_contradiction`. The document gives two incompatible death dates for Paul Leder, so hop 2 has conflicting values.

`evidence_contradiction` `ID: f7af30340bdd11eba7f7acde48001122` **(2WikiMultiHopQA)**

Query: When did the director of film Molly & Gina die?

Gold documents:

⋆ [Molly & Gina] "Molly & Gina is a 1994 movie directed by Paul Leder and written by Leder and his son, Reuben. It stars Frances Fisher, Natasha Gregson Wagner, and Peter Fonda."

⋆ [Paul Leder] "Paul Leder (March 25, 1926 – April 9, 1996) was an American film director, writer and producer. He is most famous today for his films 'A*P*E' and 'I Dismember Mama'. With wife Etyl

Leder, he is the father of modern-day producer and director Mimi Leder, writer Reuben Leder, and casting director Geraldine Leder. Paul Leder died of lung cancer on April 8, 1996 at age 70."

Official reasoning chain:

hop 1: Molly & Gina → director → *Paul Leder*

hop 2: Paul Leder → date of death → *April 9, 1996*

Official answer: April 9, 1996

Where it breaks: one document gives two death dates for the same person. Its first sentence says April 9, 1996, and its last sentence says April 8, 1996. Since a person dies only once, one of the two sentences is wrong, and the official answer takes the first date without resolving the conflict.

C.2.4 CORRUPTED_OR_UNREADABLE_EVIDENCE

A document about an entity on the chain is damaged, for example with a sentence cut off midway, text turned into garbled characters or a table flattened into one unreadable line, and a fact that the chain needs cannot be read from what remains or from any other supplied document (Figure 16). Such a sample therefore also meets the definition of `required_evidence_absent`. If the documents lack the fact but their text is intact, the sample carries `required_evidence_absent` but not this label.

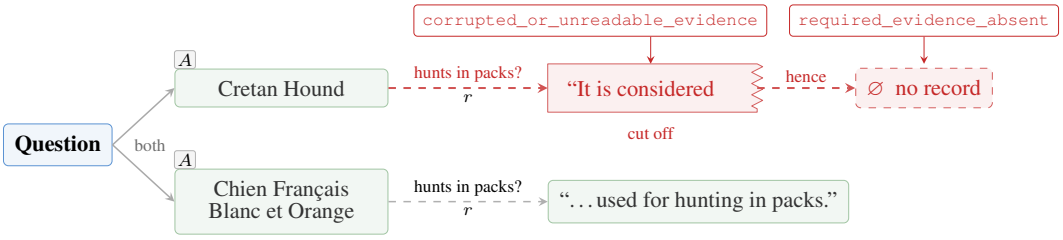


Figure 16: `corrupted_or_unreadable_evidence`. The Cretan Hound document breaks off mid-sentence, so the hop about that breed ends in damaged text and reaches no stated value. The sample therefore carries both labels.

`corrupted_or_unreadable_evidence` `ID: 5a893fc8554299515336l2f0` **(HotpotQA)**

Query: Are the Chien Français Blanc et Orange and Cretan Hound both used for hunting in packs?

Gold documents:

⋆ [Chien Français Blanc et Orange] "The Chien Français Blanc et Orange (FCI No.316) is a breed of dog of the scenthound type, originating in France. The breed is used for hunting in packs and descends from the old Hound of Saintonge type of large hunting dog."

⋆ [Cretan Hound] "The Kritikos Lagonikos (Cretan Hound) (Greek:Κρητικός Λαγωνικός) is a hunting breed of dog from the island of Crete, in Greece. It is considered"

The only non-gold document that mentions the Cretan Hound says nothing about pack hunting:

○ [Warren Hound] "... the Cretan Hound has usually rose-ears and a curly or curved tail..."

Official answer: no

Where it breaks: the question needs one fact about each breed. The first document states that the Chien Français Blanc et Orange is used for hunting in packs, but the Cretan Hound document breaks off in the middle of its second sentence, and no other document says whether the Cretan Hound is used for hunting in packs. The official answer, no, therefore rests on a fact that cannot be read in the documents.

C.3 POINTER FAMILY

Pointer-family defects sit in the gold marks rather than in the question or the document text. All the documents together can answer the question, but the text that carries the gold marks cannot. As described in Appendix A.1, one gold mark selects one sentence in HotpotQA and 2WikiMultiHopQA and one whole document in MuSiQue. A fact is needed when the chain cannot be completed without it, and a mark is useful when the text it selects states at least one needed fact and useless when it states none.

C.3.1 GOLD_EVIDENCE_INCOMPLETE

Every existing mark is useful, but at least one needed fact is stated only in text that carries no mark, so the marks alone cannot complete the chain (Figure 17). If no supplied document states the fact, the label is `required_evidence_absent`, and if some existing mark is also useless, it is `gold_evidence_mislabeled`. That two mentions of one name are the same entity is never a needed fact, since the bridge convention of the Hop family already assumes it. Leaving a sentence that states it unmarked is therefore not this defect.



Figure 17: `gold_evidence_incomplete`. Both marks are useful, but neither says which voice came first. Only unmarked documents do.

`gold_evidence_incomplete` `ID: 2hop__64175_90973` **(MuSiQue)**

Query: Who did the original voice of the character Mila Kunis plays on Family Guy?

Gold documents:

⋆ [List of Family Guy cast members] "Mila Kunis voices Meg Griffin. Kunis won the role after auditions and a slight rewrite of the character, in part due to her performance on That '70s Show…"

⋆ [Meg Griffin] "Meg Griffin Family Guy character First appearance 1998 Pilot Pitch of Family Guy … Created by Seth MacFarlane Voiced by Lacey Chabert (1999 – 2000, 2011, 2012) Mila Kunis (1999 – present) Tara Strong (singing voice)…"

Not marked as gold:

○ [Meg Griffin] "Meg made her first appearance on television when Family Guy debuted on Fox on January 31, 1999, with the episode 'Death Has a Shadow'. Originally voiced by Lacey Chabert during the first season, she has been voiced by Mila Kunis since season 2…"

Official reasoning chain:

hop 1: Mila Kunis → plays on Family Guy → *Meg Griffin*
hop 2: Meg Griffin → original voice → *Lacey Chabert*

Official answer: Lacey Chabert

Where it breaks: both gold documents are useful. The first states that Mila Kunis voices Meg Griffin, and the second names Lacey Chabert among Meg's voice actors. Together they are still not enough, because the second gives 1999 as the starting year for both Chabert and Kunis and so cannot say which voice came first. That order appears only in text without a gold mark, such as the unmarked Meg Griffin document above.

C.3.2 GOLD_EVIDENCE_MISLABELED

At least one existing mark is useless, and a needed fact is stated only in text that carries no mark, so the marks alone cannot complete the chain (Figure 18). A useless mark may fall on an unrelated sentence or document or on a fragment that states nothing. If every existing mark is useful, the label is `gold_evidence_incomplete`. A useless mark alongside marks that already complete the chain is not a defect.

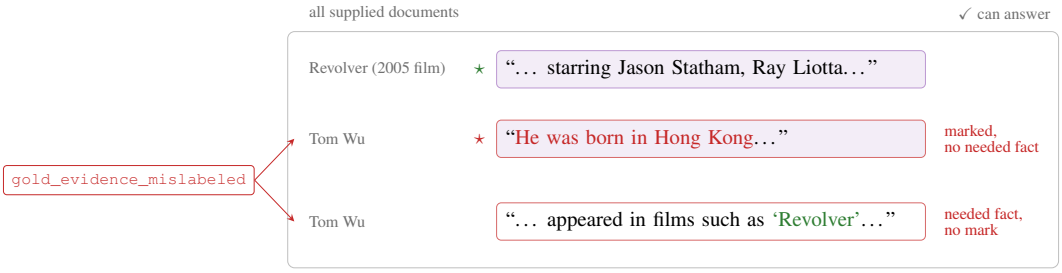


Figure 18: `gold_evidence_mislabeled`. The mark in the Tom Wu document falls on his birthplace, which the question never asks about, while the sentence that names his films carries no mark.

`gold_evidence_mislabeled` `ID: 5a7172105542994082a3e837` **(HotpotQA)**

Query: What film did Tom Wu appear in that also starred Jason Statham and Ray Liotta?

Gold documents:

[Revolver (2005 film)]
⋆ s0: "Revolver is a 2005 British-French crime thriller film co-written and directed by Guy Ritchie and starring Jason Statham, Ray Liotta, Vincent Pastore and André Benjamin."
s1: "The film centres on a revenge-seeking confidence trickster whose weapon is a universal formula…"

[Tom Wu]
s0: "Tom Wu is a Chinese British actor."
⋆ s1: "He was born in Hong Kong and grew up in Chinatown, London."
s2: "He is a martial arts expert who has appeared in films such as 'Revolver', 'Shanghai Knights', 'Batman Begins' and the Bollywood sci-fi movie 'Ra.One'."

Official answer: Revolver

Where it breaks: one mark is useful and one is not. The Revolver document is marked on the sentence that names Jason Statham and Ray Liotta, which the question needs. The Tom Wu document is marked on his birthplace, which the question never asks about. The sentence that ties him to Revolver comes right after it and carries no mark. With only the marked sentences, nothing connects Tom Wu to any film.

C.4 ANSWER FAMILY

Answer-family defects sit after the chain ends, in what the benchmark records as the answer. The chain itself is healthy and ends at one answer that the documents support, the answer entity C of Figure 9, which we call the supported answer. The benchmark records an official answer and, for some samples, a list of aliases that a scorer also accepts (Appendix A.1).

C.4.1 INCORRECT_ANSWER

The official answer and the supported answer refer to different entities or facts (Figure 19). If the two refer to the same thing in different forms, such as another spelling or a less precise date, this label does not apply, and the case may fall under `incomplete_answer_aliases` or `answer_format_or_granularity_mismatch`.

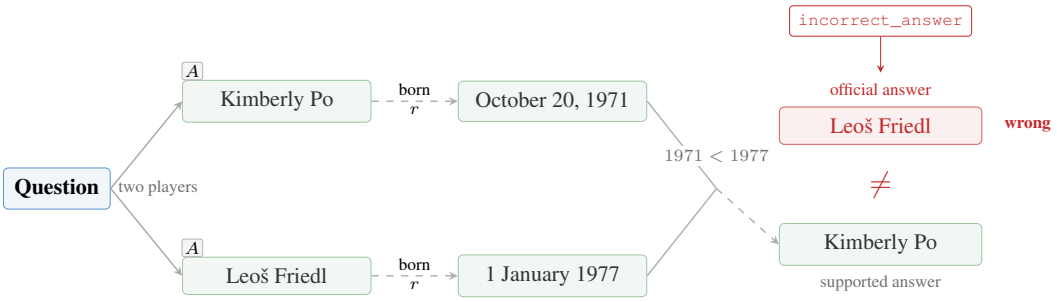


Figure 19: `incorrect_answer`. Both birth dates are stated and Po's is earlier, so the supported answer is Kimberly Po, but the official answer names the wrong player.

`incorrect_answer` ID: `5a7269fa5542992359bc308d` **(HotpotQA)**

Query: Which tennis player was born first, Kimberly Po or Leoš Friedl?

Gold documents:

⋆ [Leoš Friedl] "Leoš Friedl (born 1 January 1977 in Jindřichův Hradec) is an inactive Czech professional tennis player best known for his doubles play with František Čermák. . . "

⋆ [Kimberly Po] "Kimberly Po-Messerli (born October 20, 1971) is a former professional tennis player from the United States."

Official answer: Leoš Friedl (aliases: none)

Where it breaks: the chain is healthy. Both birth dates are stated, and comparing them gives one answer. Po was born in 1971 and Friedl in 1977, so Po was born first and the supported answer is Kimberly Po. The official answer names the wrong player, Friedl.

C.4.2 INCOMPLETE_ANSWER_ALIASES

The official answer and the supported answer refer to the same entity or fact, but the question or the documents give another equivalent form of that answer, such as a translated name or a different spelling, that neither the official answer nor its aliases cover (Figure 20). Accepting an alias that names something else is a different defect, `answer_format_or_granularity_mismatch`.

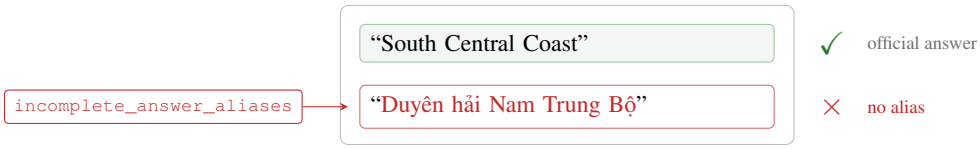


Figure 20: `incomplete_answer_aliases`. The gold document gives the region two names. The official answer covers the English name, but no alias covers the Vietnamese one.

`incomplete_answer_aliases` ID: `3hop2__607269_223623_162182` **(MuSiQue)**

Query: In what region of the country containing Hạ Hòa is John Phan's birthplace found?

Gold documents:

⋆ [Hạ Hòa District] "Hạ Hòa is a rural district of Phú Thọ Province in the Northeast region of Vietnam. . . "

⋆ [John Phan] "Bon 'John' Phan (born October 10, 1974 in Da Nang, Vietnam) is a Vietnamese-American professional poker player based in Stockton, California. . . "

⋆ [South Central Coast] "South Central Coast (Vietnamese: Duyên hải Nam Trung Bộ) is one of the regions of Vietnam. It consists of the independent municipality of Đà Nẵng and seven other provinces. . . "

Official reasoning chain:

hop 1: Hạ Hòa → country → *Vietnam*
hop 2: John Phan → place of birth → *Da Nang*
hop 3: Da Nang → region of Vietnam → *South Central Coast*

Supported answer: South Central Coast

Official answer: South Central Coast (aliases: none)

Where it breaks: the official answer is right and every hop is supported. The defect is in the alias list, which is empty, while the gold document itself gives the region a second name, Duyên hải Nam Trung Bộ. An answer that gives this Vietnamese name would be scored as wrong by string matching.

C.4.3 ANSWER_FORMAT_OR_GRANULARITY_MISMATCH

The official answer and the supported answer refer to the same entity or fact, but not everything the benchmark accepts matches the supported answer. This happens in two ways. First, the official answer gives the right thing at the wrong precision or in the wrong form, for example only the year when the question asks for a date of birth and the documents give the full date, or an exact figure when the documents give only a lower bound (Figure 21). Second, one of the aliases refers to a different entity or fact from the supported answer.

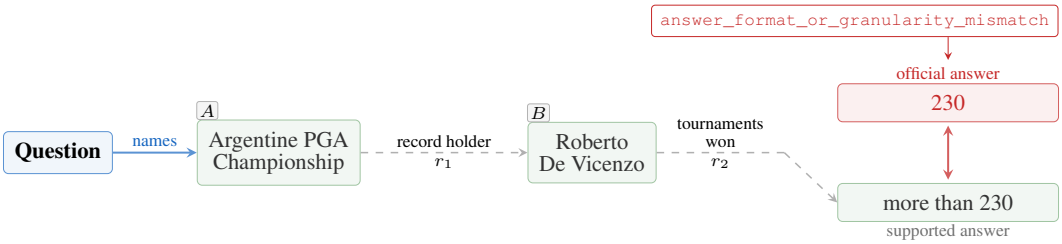


Figure 21: `answer_format_or_granularity_mismatch`. The document gives a lower bound, more than 230, but the official answer states an exact figure, 230.

`answer_format_or_granularity_mismatch` `ID: 5a887bc0554299206df2b2b6` **(HotpotQA)**

Query: The Argentine PGA Championship record holder has won how many tournaments worldwide?

Gold documents:

⋆ [Argentine PGA Championship] "The Argentine PGA Championship is a golf tournament played in Argentina since 1920. It was generally supported by the leading Argentine golfers and its list of champions includes Eduardo Romero (8 times), José Jurado (7 times), Vicente Fernández (5 times), José Cóceres (twice) and Ángel Cabrera. However, the record holder is former British Open champion Roberto De Vicenzo, who recorded 16 victories between 1944 and 1985."

⋆ [Roberto De Vicenzo] "Roberto De Vicenzo (14 April 1923 – 1 June 2017) was a professional golfer from Argentina. He won more than 230 tournaments worldwide in his career including eight on the PGA Tour and most famously the 1967 Open Championship."

Supported answer: more than 230

Official answer: 230 (aliases: none)

Where it breaks: the chain is healthy. The first document names Roberto De Vicenzo as the record holder, and the second says he won more than 230 tournaments. The defect is the official answer's precision, since it drops the qualifier and states the exact figure 230.

D THE LLM AUDITOR

This appendix describes the three calls of the LLM Auditor of Section 4.2.1, its setup, the passages of its prompts that set the standard of the audit and the automatic checks applied to every audit.

D.1 THE THREE CALLS AND THEIR SETUP

Analyzer The analyzer receives the complete sample, including the question and the official answer with its aliases. It also receives every document the sample ships with, gold and distractor alike, with the benchmark's marks shown on the gold ones. It drafts the initial audit. The distractors are included deliberately, because the analyzer judges the whole record as well as the marked part of it. The unmarked text may hold the evidence a hop needs, or a fact that contradicts what the gold documents say.

Critic The critic first decides on its own whether the sample's question can be answered from its gold documents, and only then judges the analyzer's initial audit, hunting missed defects and false accusations alike. Deciding first is what stops it from anchoring on the analyzer's conclusion. When it agrees, the initial audit stands as the *final audit* and the sample is done after two LLM calls. When it objects, it must state specific, quoted objections.

Revision A single revision call follows only when the critic objects. It receives the sample, the initial audit and the objections, and is instructed to re-evaluate the sample rather than defer to the critic; it writes a complete replacement, which becomes the final audit. There is no further round, so a sample that the critic objects to goes through exactly three calls. The critic agreed with the initial audit on 90.7% of the samples, so only the remaining 9.3% needed a revision.

The analyzer, the critic and the revision all ran on GPT-5.6 (`gpt-5.6-sol`) through the OpenAI Agents SDK (Responses API), with reasoning effort set to high. Each call works on one sample without tools or web access, returns an object in a fixed output format rather than free text and is attempted at most three times. An output that fails a check of Appendix D.3 is sent back to the model with the violation quoted, and a call whose output is cut off before its format is complete is made again with a larger output-token budget and less reasoning. 99.2% of the audits ran at high reasoning effort throughout, and the other 0.8% needed such a retry.

D.2 PROMPTS

The three calls share one rulebook, and each adds a short prompt of its own. The box below quotes, in their original order, the passages of the rulebook that set the standard every audit has to meet. The rulebook's `decomposition` is the field that holds the official reasoning chain of Appendix A.1.

Shared rulebook (excerpts)

The central question is:

Given the original question and the official gold evidence, can a capable LLM derive the official answer?

The official `decomposition` may be used only as a non-authoritative signpost for the dataset's intended entities, relations, and reasoning direction. It is not evidence and every factual claim suggested by it must still be established by the gold evidence.

A wrong or incomplete decomposition is not itself a dataset error for this audit, must not produce an error category, and must not override the meaning of the original question.

Audit two evidence layers separately:

1. Gold-evidence layer: decide whether the original question plus official gold evidence supports the official answer.
2. Full-record layer: inspect all supplied paragraphs only to distinguish an incomplete/mislabeled gold annotation from evidence absent from the record.

Most records may be valid. Do not create an error merely because wording is awkward, evidence is concise, a fact is paraphrased, or a reasonable deductive step connects facts that are explicitly stated. Conversely, do not accept plausible outside knowledge as evidence.

Preserve the original inference burden whenever the supplied evidence permits. Do not turn a multi-step inference question into a direct fact lookup merely to make it answerable. Do not add intermediate

answers, explanatory appositives, parenthetical hints, copied evidence clauses, or extra entity descriptions that reveal the reasoning path or make retrieval unnecessary.

D.3 Automatic Checks

Code checks every audit and every critic review before accepting it. Code also derives the final fields of each sample.

Citations Every quotation, in an audit and in a critic's objection alike, must appear in the source it declares, ignoring differences in whitespace and letter case. Documents are cited by the sample's own indices, and a sentence index may be given only when it names one of that document's gold sentences.

Consistency An audit is rejected when its parts cannot all hold at once, for example when it judges the gold evidence fully supporting but assigns a label under which the gold evidence cannot support the official answer, such as `gold_evidence_incomplete`, or assigns `required_evidence_absent` to a sample it judges answerable from its documents.

Repairs A proposed repair may change only the fields that its labels permit, except that the alias list may be changed under any label. Every gold sentence it adds must appear in an existing document, again ignoring whitespace and letter case, and the word count of its replacement question may differ from that of the original by at most $\max(3, \lceil 0.2\, n \rceil)$, where n is the word count of the original.

Derived fields The verdict and the primary label of a sample are computed by code from the labels of its final audit and are never written by a model. A sample whose final audit has no label is clean; for any other sample, the primary label is the first of its labels in a fixed priority order. The order runs from the labels under which the question cannot be answered from its documents, the Query and Hop families, through those under which the benchmark records wrong evidence or a wrong answer, `incorrect_answer` and the Pointer family, to those that affect only scoring, `incomplete_answer_aliases` and `answer_format_or_granularity_mismatch`; within each group the labels keep their order in Figure 3.

D.4 Agreement with the Blind Annotation

Table 8 compares the LLM Auditor with the blind annotation of Section 5.2. The LLM Auditor's binary verdict on a sample is the decision the repaired benchmarks release, so the one HotpotQA sample with no usable audit counts as defective, since they remove it (Section 5.2). Having no defect labels, it is left out of the label comparison. We report Cohen's κ and Gwet's AC1 (Gwet, 2008) side by side for the reason given in Appendix E.4, and scored the LLM Auditor as a detector of the defective class against the blind labels. On the defect labels we compared the primary label, the first of a sample's labels in the priority order of Appendix D.3, and only on the samples both sides called defective. The binary verdict shows that the two sides draw the line between clean and defective in much the same place, and the defect labels show that on most defective samples they also name the same defect.

Table 8: Agreement between the LLM Auditor and the blind annotation of the 300 hand-audited samples. Precision, recall and F1 score the LLM Auditor against the blind labels.

		Binary agreement			Defective-class detection			Defect-label agreement		
Benchmark	Samples	Raw agr.	Cohen's κ	Gwet's AC1	Precision	Recall	F1	Raw agr.	Cohen's κ	Gwet's AC1
HotpotQA	100	0.780	0.497	0.610	0.724	0.600	0.656	0.650	0.560	0.606
2WikiMultiHopQA	100	0.920	0.787	0.872	0.808	0.875	0.840	0.714	0.498	0.685
MuSiQue	100	0.790	0.566	0.595	0.875	0.778	0.824	0.653	0.489	0.625
All	300	0.830	0.643	0.676	0.820	0.746	0.781	0.667	0.541	0.638

E THE HUMAN VERIFICATION PROTOCOL

E.1 SAMPLING AND BATCH CONSTRUCTION

The draw The verification pool is 1,000 samples per benchmark, 3,000 in all, drawn by simple random sampling from each development set under a fixed seed, so the draw is reproducible.

The six batches The 3,000 samples were partitioned, again under a fixed seed, into six batches of exactly 500 with no overlap and no sample left out. Each batch holds 166 or 167 samples from each benchmark, so that every annotator receives the same mix of benchmarks and a comparable workload and difficulty.

The assignment The six batches were assigned to the six annotators, P1–P6, in two rounds (Figure 4, bottom). In the first round, batch k went to annotator Pk. In the second round, each batch went to the annotator before its first one, batch 1 to P6 and every other batch k to P$(k-1)$, so no annotator judged the same sample twice.

E.2 SAMPLE SIZES IN PRIOR WORK

Table 9 lists recent works that evaluate on 1,000 samples per benchmark of HotpotQA, 2WikiMultiHopQA and MuSiQue, the scale at which our verified benchmark was drawn (Section 4.2.2). All seven draw their samples from the development sets, and two of them state that the draw is random.

Table 9: Works that evaluate on 1,000 samples per benchmark of HotpotQA, 2WikiMultiHopQA and MuSiQue, as stated in the experimental setup of each paper.

Work	Venue	Samples per benchmark	Sampling
HippoRAG (Gutierrez et al., 2024)	NeurIPS 24	1,000	not stated
SiReRAG (Zhang et al., 2025b)	ICLR 25	1,000	not stated
HippoRAG 2 (Gutiérrez et al., 2025)	ICML 25	1,000	random
LogicRAG (Chen et al., 2026a)	AAAI 26	1,000	not stated
ASearcher (Gao et al., 2025)	NeurIPS 25 Workshop	1,000	random
LinearRAG (Zhuang et al., 2026)	ICLR 26	1,000	not stated
MemGraphRAG (Wu et al., 2026a)	KDD 26	1,000	not stated

E.3 REVIEWING ONE SAMPLE

This subsection describes how an annotator reviews one sample in the verification of Section 4.2.2. The annotator starts from the final audit of the LLM Auditor, confirms or overturns its verdict, corrects its labels and accepts or rewrites its repair. The repaired benchmarks apply the LLM Auditor's own repairs and do not depend on this procedure. Annotators judge every sample by the label definitions and boundary rules of Appendix C.

Step 1: Decide whether the sample is defective. The annotator reads the question, the official answer and the supplied documents, with the gold marks shown, and then the LLM Auditor's audit of them, including its conclusion, its labels and its reasoning. Only the final audit is shown, not the initial audit or the critic's objections. Neither answer to Q1 is pre-selected, so the first judgment is the annotator's own.

Q1: Is this sample defective?

- No → the sample is `clean` and the judgment is done, whatever the LLM Auditor concluded.
- Yes → go to Step 2.

Step 2: Check whether the labels and the reasoning are correct. The annotator checks whether the quoted sentences support what the LLM Auditor claims and whether the assigned labels match

the defect it describes. If the LLM Auditor called the sample clean, it gave no labels to accept, so the annotator answers No to Q2 and selects the labels.

Q2: Are the assigned labels and the reasoning behind them correct?

- Yes → accept the diagnosis and go to Step 3; the LLM Auditor's labels and the explanation it gave for each are recorded as the annotation's own.
- No, the defect is real but the labels or the reasoning are wrong → select the labels that do apply, add one plain sentence for each explaining why it applies, and go to Step 3.

Step 3: Decide whether and how the sample can be repaired. When the LLM Auditor proposed a repair, the annotator starts from that proposal, a replacement question, answer and alias list plus any changes to the gold marks. Otherwise, whether the LLM Auditor called the sample clean or proposed no repair, the annotator starts from the original sample.

Q3: Can the sample be repaired?

- The proposed repair is good → accept it; `fixable`, repaired by the LLM Auditor.
- The proposed repair is not good, or none was given, but the annotator can repair the sample → write the replacement; `fixable`, repaired by the annotator.
- Neither the proposal nor the annotator can produce a working replacement → `unfixable`, and the sample is dropped.

A repair is good when the repaired question is readable and its gold documents support its answer. It must keep the original reasoning burden, as the rulebook of the LLM Auditor also requires (Appendix D.2), and must not leak the answer into the question. Only the question, the answer, the alias list and the gold marks may change, and the text of the documents is never edited.

Q4: How many reasoning hops does the sample take before and after the repair? For every `fixable` sample the annotator judges both hop counts and records them (reported in Table 21).

Every judgment also carries a confidence rating and the number of minutes the annotator reports spending on the sample.

Final output. Every sample ends with exactly one outcome, `clean`, `fixable` or `unfixable`. A `fixable` or `unfixable` sample also records which labels apply and why, and a `fixable` sample carries a complete replacement of its question, answer, alias list and gold marks, either the LLM Auditor's proposal that the annotator accepted or one the annotator wrote. Section 4.2.2 describes how the two judgments of each sample decide whether it enters the verified benchmark.

Outcomes in Figure 6. The right block of Figure 6 places each reviewed sample in exactly one column: *Accepted*, both annotators kept the LLM Auditor's verdict and, if it proposed a repair, the verified benchmark keeps that repair unchanged; *Rewritten*, both kept its verdict but the first-round annotator rewrote its repair; *Removed*, both judged the sample `unfixable`; *Overlooked*, the LLM Auditor had called the sample clean but both annotators judged it `fixable`; *Overturned*, the LLM Auditor had called the sample defective but both annotators judged it `clean`; *Disputed*, the two annotators gave different judgments. *Total* counts the samples that enter the verified benchmark, all but the removed and the disputed ones. In the left block, *Unchanged* counts the samples released with all four fields identical to the original, including 11 whose proposed repair of the gold marks left them as they were, and *Removed* instead counts the samples for which the LLM Auditor produced no usable repair (Section 5.2). These include 36 samples for which it returned no usable output, neither labels nor a repair, so the defect rates of Figure 5 are computed over the other 22,362 samples.

E.4 Agreement between Annotators

The two annotators of a sample agree closely, both on whether it is defective and on whether it can be repaired. Every sample of the verification pool was judged in both rounds of Appendix E.1, each time by a different annotator who did not see the other's judgment. Table 10 reports the agreement between the two judgments, as raw agreement, Cohen's κ and Gwet's AC1 (Gwet, 2008), on the binary verdict, clean or defective, and on the three-way judgment, `clean`, `fixable` or `unfixable`. We report Cohen's κ and Gwet's AC1 side by side because the two classes are unevenly balanced in

every benchmark, which depresses κ but affects AC1 much less, so reading the two together keeps the effect of this imbalance from being mistaken for weak agreement. On the binary verdict the two annotators reach a raw agreement of 0.944, a Cohen's κ of 0.884 and a Gwet's AC1 of 0.891, almost perfect agreement in the bands of Landis & Koch (1977), and κ is at least 0.866 on every benchmark. For 91.6% of the 1,155 samples that both annotators called defective, they also agree on whether the sample is `fixable` or `unfixable`. Taken together, the two decisions give a three-way raw agreement of 0.911 and a Cohen's κ of 0.823, also almost perfect agreement. Only samples on which the two annotators give the same judgment can enter the verified benchmark (Section 4.2.2), so each of its samples carries a judgment that does not hinge on which annotator made it.

Table 10: Agreement between the two annotators on the samples judged in both rounds. The binary verdict is clean or defective, and the three-way judgment is `clean`, `fixable` or `unfixable`.

		Binary verdict			Three-way judgment		
Benchmark	Samples	Raw agr.	Cohen's κ	Gwet's AC1	Raw agr.	Cohen's κ	Gwet's AC1
HotpotQA	1,000	0.939	0.868	0.887	0.921	0.832	0.897
2WikiMultiHopQA	1,000	0.947	0.866	0.912	0.925	0.814	0.906
MuSiQue	1,000	0.945	0.885	0.895	0.888	0.782	0.849
All	3,000	0.944	0.884	0.891	0.911	0.823	0.882

F Re-evaluation Setup

This appendix lists the nine re-evaluated methods (Table 11), describes how each one was run and gives the protocol used to score them.

F.1 The Nine Methods

Table 11: The nine re-evaluated RAG methods, three per family, each with its official code.

Category	Method	Venue	Code
Retrieval structure	LinearRAG	ICLR 26	`github.com/DEEP-PolyU/LinearRAG`
	GraphRAG	arXiv 24	`github.com/microsoft/graphrag`
	HippoRAG 2	ICML 25	`github.com/OSU-NLP-Group/HippoRAG`
Workflow	LogicRAG	AAAI 26	`github.com/chensyCN/LogicRAG`
	STRIDE	SIGIR 26	`github.com/MINE-USTC/STRIDE`
	IRCoT	ACL 23	`github.com/StonyBrookNLP/ircot`
Agentic search	Search-R1	COLM 25	`github.com/PeterGriffinJin/Search-R1`
	ReSearch	NeurIPS 25	`github.com/Agent-RL/ReCall (branch re-search)`
	Search-o1	EMNLP 25	`github.com/RUC-NLPIR/Search-o1`

F.2 Implementation

Shared setup Every method retrieved from a closed pool, which for each benchmark is a corpus of that benchmark's own paragraphs with the gold marks stripped. Retrieval depth is k=5 documents per query. IRCoT, Search-R1, ReSearch and Search-o1 shared one BM25 index per benchmark, and the other methods used the retriever that their own code defines (Table 12). Methods that do not release a trained model of their own used Qwen3-32B, served locally, as the backbone. We did not tune any method on the benchmarks, since tuning would mean evaluating a variant of the method rather than the published system.

Departures from the shared setup For Search-R1 and ReSearch, which are trained with reinforcement learning, the checkpoint is the method, whereas for the other methods it is the pipeline around the model. The two ran their own released checkpoints, since swapping in the shared backbone would mean evaluating a different system. GraphRAG fills a token budget from an index it

Table 12: Retriever, index construction and backbone of each method. IRCoT, Search-R1, ReSearch and Search-o1 share one BM25 index per benchmark.

Method	Retriever	Index construction	Backbone
Retrieval structure			
LinearRAG	BFS over an entity–passage graph	spaCy NER (en_core_web_trf); all-mpnet-base-v2 embeddings	Qwen3-32B
GraphRAG	Local search over its own graph	Graph and community summaries built with Qwen3-32B; bge-m3 embeddings	Qwen3-32B
HippoRAG 2	Personalized PageRank over its knowledge graph	OpenIE with Qwen3-32B; NV-Embed-v2 embeddings	Qwen3-32B or GPT-5.6
Workflow			
LogicRAG	Dense retrieval	all-MiniLM-L6-v2 embeddings	Qwen3-32B
STRIDE	Dense retrieval	Contriever embeddings in a FAISS index	Qwen3-32B
IRCoT	BM25	Shared BM25 index	Qwen3-32B or GPT-5.6
Agentic search			
Search-R1	BM25	Shared BM25 index	Released checkpoint SearchR1-nq_hotpotqa_train-qwen2.5-7b-em-ppo (SearchR1-7B in our tables)
ReSearch	BM25	Shared BM25 index	Released checkpoint ReSearch-Qwen-32B-Instruct (ReSearch-32B in our tables)
Search-o1	BM25	Shared BM25 index	Qwen3-32B or GPT-5.6

builds itself rather than retrieving a fixed number of documents, so it has no k to unify. We left this budget at the default its code ships with, 12,000 tokens per question. In our runs, depending on the benchmark, this budget held a median of 34 to 41 text units, the corpus chunks that GraphRAG indexes. Almost every text unit was a whole document, while the other methods saw five documents per query.

Short answers from GraphRAG GraphRAG is built for query-focused summarization, so, as released, it answers in multiple paragraphs with inline citations rather than with the short answer that the three benchmarks expect. We therefore ran it as HippoRAG 2 does in its evaluation of GraphRAG, with GraphRAG's own `response_type` parameter set to "Short phrase" and the response rephrased into a short answer by Qwen3-32B with HippoRAG 2's QA prompt. HippoRAG 2 releases this prompt but not the code that runs this step, so we took the prompt verbatim from its repository and implemented the step from the description in its paper. Beyond this, GraphRAG's own code was left as released.

GPT-5.6 as the backbone For HippoRAG 2, IRCoT and Search-o1 we also ran every LLM call at query time on GPT-5.6 (`gpt-5.6-sol`) at reasoning effort `xhigh` with up to 64,000 output tokens, and left the index and the retriever unchanged. We ran GPT-5.6 on one method per family, the one that scores highest in its family among the methods whose backbone can be replaced, by LLM-judge accuracy on Qwen3-32B averaged over the full development sets of the three original benchmarks (Table 3). In agentic search this leaves only Search-o1, since Search-R1 and ReSearch are tied to their own checkpoints. The GPT-5.6 API does not accept a stop parameter, so we cut each reply at the first stop string the method requests. Search-o1 marks its searches with special tokens written into the text, which GPT-5.6 does not reliably produce, so we offered its search to GPT-5.6 as a function tool and translated each call back into the method's own format.

F.3 Evaluation Metrics

We report four answer-level metrics for every method and every benchmark version. Each compares a prediction with the answer that the version records, which is the official answer on the original benchmark and the repaired answer wherever a sample was repaired. **EM** counts a prediction correct when, after SQuAD-style normalization, it equals the answer or one of its aliases. **Cover-EM** also accepts a prediction that contains the answer, and **F1** is the best token-overlap F1 over the answer and its aliases. All three compare strings and so penalize a right answer that is worded differently, which is what the fourth metric, the LLM judge, is for.

The LLM judge follows the zero-shot, context-free protocol of Ho et al. (2026), with Llama-3.3-70B, a model from a different family than any backbone we evaluated, as the judge. Scoring runs as an EM-first cascade, in which an exact match is accepted outright, an empty prediction is rejected and every other prediction is judged correct or incorrect against the same answer. The reported score is the fraction of predictions that are an exact match or judged correct. In the study this protocol comes from, the Llama-3.3-70B judge agrees with human judgments far more closely than string matching does, with an average Pearson correlation of 0.750 in its initial setting that rises to 0.819 without examples and to 0.820 without context, against 0.220 for EM and 0.404 for F1.

On the repaired and verified benchmarks, a sample whose repair rewrites the question was answered again by every method on the new question, while a sample whose repair changes only the answer, the aliases or the gold marks kept its original prediction and was scored against the repaired answer. In the Oracle setting the model reads each question together with its gold documents, so a sample whose repair changes which documents are gold was answered again as well. A removed sample leaves the benchmark, so the repaired benchmarks hold 7,374, 12,500 and 2,381 samples and the verified benchmark 920, 924 and 888. A sample on which a method returns no answer counts as wrong, including an answer that the provider's content filter cuts off.

G Re-evaluation Results in Full

G.1 Full Results on the Two Released Benchmarks under Four Metrics

Table 13 scores every method under the four metrics of Appendix F.3 on the original and the repaired benchmarks over the full development sets, and Table 14 on the original and the verified benchmarks over the 1,000 samples per benchmark (Appendix E.1). The original columns score all of these samples as published, with none removed. The repaired benchmarks leave out the samples for which the LLM Auditor produced no usable repair (Section 5.2), and the verified benchmark leaves out the disputed samples and those that both annotators judged `unfixable` (Section 4.2.2), so the original columns hold more samples than the released columns, as n under each column shows. The Oracle rows follow the setting of Section 2 and Appendix A.4. The last value in each cell is the LLM-judge accuracy in Table 3, and Δ is its change from the original. Next to each Δ, in brackets, is its 95% confidence interval from a paired bootstrap (Koehn, 2004) with 10,000 resamples of these samples. In each resample, a sample that the released benchmark leaves out counts only toward the original score, as in Table 3.

The gains in Table 3 hold under every metric and lie beyond sampling noise. Every method and Oracle setting scores higher on the released benchmarks under all four metrics and on all three benchmarks. The 95% interval of every LLM-judge gain lies above zero, and even the smallest gain, that of Search-R1 on 2WikiMultiHopQA, has an interval starting at 1.6 points on the repaired benchmark and 0.5 on the verified benchmark. Because every method was scored on the same resamples, the bootstrap also bounds the difference between the gains of two methods. On the full MuSiQue development set, HippoRAG 2 gains 5.1 points more than LinearRAG and 2.1 more than IRCoT, with intervals of 3.0 to 7.1 and 0.1 to 4.1, and ReSearch gains 3.6 points more than Search-R1, with an interval of 1.5 to 5.7, so the defects do cost different methods different amounts, as Section 6.2 reports.

The original benchmarks therefore understate every method whichever metric is used, and the gains in Table 3 are not an artifact of sampling. Appendix G.2 shows that the gains do not come from removing samples either.

Table 13: All four metrics on the original and the repaired benchmarks. Each cell reads EM | Cover-EM | F1 | LLM judge, and Δ is the change in LLM-judge accuracy from original to repaired, with its 95% confidence interval in brackets.

Method	Base LLM	HotpotQA original $n = 7{,}405$	HotpotQA repaired $n = 7{,}374$	HotpotQA Δ	2WikiMultiHopQA original $n = 12{,}576$	2WikiMultiHopQA repaired $n = 12{,}500$	2WikiMultiHopQA Δ	MuSiQue original $n = 2{,}417$	MuSiQue repaired $n = 2{,}381$	MuSiQue Δ
Retrieval structure										
LinearRAG	Qwen3-32B	25.7|50.2|40.6|62.2	34.3|60.5|46.9|64.9	+2.7 [2.1, 3.3]	17.0|48.6|32.6|47.0	21.5|54.1|37.6|52.2	+5.2 [4.7, 5.7]	10.8|24.0|20.1|35.3	19.3|35.6|29.7|44.4	+9.1 [7.5, 10.9]
GraphRAG	Qwen3-32B	49.3|58.1|63.7|76.4	63.4|70.2|72.9|80.5	+4.1 [3.5, 4.6]	50.2|56.6|58.1|60.2	57.6|63.0|65.0|66.0	+5.8 [5.4, 6.4]	23.7|30.5|34.2|44.4	38.7|45.9|49.2|56.2	+11.8 [10.1, 13.5]
HippoRAG 2	Qwen3-32B	57.9|68.3|73.0|86.3	74.7|82.6|83.5|90.6	+4.3 [3.8, 4.8]	46.8|52.8|53.2|55.9	55.0|60.0|61.0|62.6	+6.7 [6.1, 7.2]	30.2|40.8|42.9|55.8	51.0|60.2|61.7|70.0	+14.2 [12.5, 15.9]
HippoRAG 2	GPT-5.6	56.8|74.1|76.2|93.2	72.7|88.3|86.4|97.3	+4.1 [3.6, 4.6]	60.9|86.2|78.5|93.0	70.0|90.6|85.1|96.7	+3.7 [3.3, 4.1]	43.7|60.4|60.5|74.1	61.1|76.0|76.0|85.9	+11.8 [10.3, 13.5]
Workflow										
LogicRAG	Qwen3-32B	47.9|55.1|61.9|74.9	61.0|66.6|70.5|78.1	+3.2 [2.7, 3.8]	59.2|68.5|68.2|71.5	66.4|74.2|74.9|76.8	+5.3 [4.7, 5.8]	26.5|35.3|38.1|49.2	41.6|49.2|52.3|59.8	+10.6 [8.9, 12.4]
STRIDE	Qwen3-32B	49.3|60.0|62.8|75.8	62.4|71.4|71.6|79.4	+3.6 [3.0, 4.2]	56.9|69.8|65.7|69.4	64.6|75.6|72.7|75.1	+5.7 [5.2, 6.2]	28.6|40.5|39.3|52.0	43.4|53.7|53.3|61.8	+9.8 [8.0, 11.6]
IRCoT	Qwen3-32B	56.8|66.4|71.4|84.4	72.5|79.4|81.2|88.1	+3.7 [3.2, 4.3]	63.8|72.1|73.5|81.3	73.5|79.7|82.0|88.7	+7.4 [6.9, 7.9]	36.3|43.6|48.0|57.8	52.0|58.5|62.5|69.9	+12.1 [10.4, 13.8]
IRCoT	GPT-5.6	63.2|71.5|78.5|91.4	81.2|86.6|89.4|95.9	+4.5 [4.0, 5.0]	74.1|84.0|84.4|94.6	83.2|89.0|91.2|98.5	+3.9 [3.6, 4.3]	53.7|63.7|66.0|75.6	72.9|79.3|81.4|87.6	+12.0 [10.5, 13.5]
Agentic search										
Search-R1	SearchR1-7B	48.1|52.1|60.0|69.1	60.5|62.8|67.7|71.7	+2.6 [1.9, 3.1]	26.3|28.2|30.4|30.5	29.3|30.6|33.2|32.5	+2.0 [1.6, 2.5]	20.5|22.8|27.8|31.7	31.9|33.7|38.9|41.5	+9.8 [8.2, 11.6]
ReSearch	ReSearch-32B	57.1|61.8|70.8|83.2	73.1|75.9|80.9|87.5	+4.3 [3.7, 4.9]	78.0|81.1|82.3|85.3	86.3|88.5|89.9|91.6	+6.3 [5.8, 6.8]	38.4|42.4|47.9|55.7	55.7|58.8|63.8|69.1	+13.4 [11.8, 15.1]
Search-o1	Qwen3-32B	50.8|56.5|65.2|79.8	65.6|70.1|75.0|83.9	+4.1 [3.5, 4.6]	68.7|78.7|77.8|81.7	77.5|85.4|85.5|88.0	+6.3 [5.8, 6.8]	28.4|33.9|39.2|52.2	43.2|47.5|52.9|62.9	+10.7 [8.9, 12.5]
Search-o1	GPT-5.6	41.5|68.9|63.9|89.3	54.2|82.5|73.2|93.3	+4.0 [3.6, 4.6]	40.7|88.9|69.3|92.1	47.5|93.1|75.0|95.9	+3.8 [3.4, 4.1]	31.6|52.8|50.0|66.8	45.6|69.3|65.2|80.3	+13.5 [11.9, 15.2]
Oracle Settings										
Oracle	Qwen3-8B	54.0|73.0|71.6|91.3	70.6|87.3|82.0|95.0	+3.7 [3.3, 4.3]	64.4|79.9|74.6|82.3	76.8|90.3|85.9|91.9	+9.6 [9.0, 10.1]	37.3|57.4|50.4|69.0	60.1|76.9|71.7|84.6	+15.6 [13.9, 17.3]
Oracle	Qwen3-32B	59.5|74.9|76.6|92.4	76.4|89.6|87.0|96.6	+4.2 [3.8, 4.8]	71.6|85.5|80.7|87.3	84.1|96.4|92.1|97.5	+10.2 [9.7, 10.8]	46.0|62.0|57.9|70.7	66.5|80.8|76.8|86.5	+15.8 [14.2, 17.4]
Oracle	Qwen3-235B	62.2|74.3|78.2|93.4	78.8|88.6|88.2|96.9	+3.5 [3.1, 4.0]	75.4|86.4|83.0|88.7	87.1|96.1|93.3|97.6	+8.9 [8.3, 9.4]	51.7|66.0|63.5|76.5	70.6|83.5|80.6|90.0	+13.5 [12.0, 15.1]
Oracle	GPT-5.6	52.1|80.5|73.4|94.6	67.0|93.6|82.9|98.7	+4.1 [3.6, 4.5]	62.7|86.5|74.1|87.7	75.6|98.7|86.3|99.3	+11.6 [11.1, 12.2]	54.8|82.1|73.3|89.3	70.7|95.2|86.8|98.5	+9.2 [8.1, 10.4]

Table 14: All four metrics on the verified benchmark and on the 1,000 samples per benchmark it is built from, as published (Appendix E.1). Each cell reads EM | Cover-EM | F1 | LLM judge, and Δ is the change in LLM-judge accuracy from original to verified, with its 95% confidence interval in brackets.

Method	Base LLM	HotpotQA original $n = 1{,}000$	HotpotQA verified $n = 920$	HotpotQA Δ	2WikiMultiHopQA original $n = 1{,}000$	2WikiMultiHopQA verified $n = 924$	2WikiMultiHopQA Δ	MuSiQue original $n = 1{,}000$	MuSiQue verified $n = 888$	MuSiQue Δ
Retrieval structure										
LinearRAG	Qwen3-32B	26.1|52.1|41.6|62.3	35.4|62.7|48.6|66.5	+4.2 [2.4, 6.0]	17.3|49.9|32.4|47.6	20.2|55.4|35.9|52.3	+4.7 [3.0, 6.4]	10.8|24.1|20.5|34.7	20.8|38.5|32.1|46.6	+11.9 [9.1, 14.9]
GraphRAG	Qwen3-32B	51.6|60.2|64.9|76.5	66.2|72.9|75.0|81.6	+5.1 [3.4, 6.9]	50.7|55.7|57.5|59.0	58.2|62.8|64.6|65.8	+6.8 [4.9, 8.8]	24.0|30.5|34.4|44.3	40.3|47.6|50.5|57.2	+12.9 [10.0, 15.9]
HippoRAG 2	Qwen3-32B	59.9|70.3|74.2|85.4	76.4|84.0|84.8|91.0	+5.6 [4.0, 7.2]	48.6|54.6|54.7|57.2	56.7|62.4|63.0|65.0	+7.8 [6.0, 9.8]	30.1|40.8|43.0|55.5	51.4|61.3|62.8|70.5	+15.0 [12.1, 17.9]
HippoRAG 2	GPT-5.6	57.3|75.2|76.3|91.6	73.4|89.9|86.9|97.0	+5.4 [3.9, 6.9]	62.0|87.1|78.5|93.0	70.7|91.7|85.0|97.2	+4.2 [2.7, 5.7]	43.6|60.7|60.4|74.6	60.6|76.6|76.3|86.5	+11.9 [9.3, 14.5]
Workflow										
LogicRAG	Qwen3-32B	49.6|57.0|62.2|73.1	63.3|68.2|71.3|77.3	+4.2 [2.5, 5.9]	59.9|68.1|68.2|70.5	68.0|75.4|75.6|77.1	+6.6 [4.7, 8.5]	26.6|36.1|38.4|50.3	43.0|50.8|53.5|62.0	+11.7 [8.7, 14.7]
STRIDE	Qwen3-32B	52.3|62.8|64.1|75.7	65.4|73.8|73.6|80.4	+4.7 [2.9, 6.6]	57.6|70.7|66.3|71.3	65.6|77.2|73.1|76.7	+5.4 [3.6, 7.3]	29.0|40.9|39.8|53.0	45.4|56.4|55.4|63.9	+10.9 [7.9, 13.8]
IRCoT	Qwen3-32B	57.9|68.0|72.3|83.7	73.8|80.5|82.1|87.9	+4.2 [2.7, 5.9]	63.7|71.9|73.4|81.2	72.8|79.2|81.2|88.2	+7.0 [5.3, 8.8]	35.7|42.5|47.8|57.8	53.2|59.0|63.8|70.8	+13.0 [10.2, 15.9]
IRCoT	GPT-5.6	63.8|72.3|79.1|90.3	81.6|87.5|89.7|95.3	+5.0 [3.6, 6.5]	75.3|84.2|85.1|94.6	83.4|89.0|91.5|98.7	+4.1 [2.8, 5.4]	54.6|64.7|67.1|77.1	73.5|79.8|82.4|88.9	+11.8 [9.4, 14.2]
Agentic search										
Search-R1	SearchR1-7B	49.8|53.8|61.1|67.7	62.8|65.0|69.2|71.2	+3.5 [1.6, 5.3]	26.9|28.6|30.6|31.5	30.0|31.7|33.9|33.8	+2.3 [0.5, 4.0]	18.8|20.9|26.5|30.7	33.2|34.3|40.1|42.0	+11.3 [8.5, 14.1]
ReSearch	ReSearch-32B	57.9|63.1|71.1|82.0	74.8|78.2|82.3|88.3	+6.3 [4.5, 8.1]	77.3|80.4|81.8|84.5	85.7|87.4|89.2|91.0	+6.5 [4.8, 8.3]	38.0|42.1|47.6|54.3	56.8|59.7|64.1|69.0	+14.7 [11.8, 17.6]
Search-o1	Qwen3-32B	51.9|56.4|65.6|77.6	68.0|71.3|77.0|82.9	+5.3 [3.6, 7.1]	72.0|80.7|79.7|83.1	80.6|87.3|87.1|89.8	+6.7 [5.0, 8.5]	29.0|34.8|40.0|52.6	47.3|50.8|56.2|66.1	+13.5 [10.5, 16.5]
Search-o1	GPT-5.6	41.4|69.1|63.3|87.7	52.4|81.8|71.4|91.6	+3.9 [2.5, 5.4]	39.8|91.1|68.5|93.1	44.8|95.1|73.5|96.6	+3.5 [2.2, 4.9]	31.5|51.6|49.8|65.7	43.4|67.7|62.9|78.6	+12.9 [10.2, 15.7]
Oracle Settings										
Oracle	Qwen3-8B	56.4|76.1|73.9|92.1	72.8|88.7|83.7|95.5	+3.4 [2.1, 4.8]	66.1|80.2|75.0|81.8	77.7|90.0|85.7|91.5	+9.7 [7.7, 11.6]	37.1|57.1|50.2|68.6	60.6|77.5|72.0|84.9	+16.3 [13.4, 19.1]
Oracle	Qwen3-32B	60.6|77.5|77.7|92.7	77.8|91.7|87.9|97.1	+4.4 [3.0, 5.7]	73.5|86.0|81.8|87.5	85.4|96.4|92.8|97.6	+10.1 [8.2, 12.0]	44.8|61.9|57.5|70.6	66.9|82.4|77.6|87.8	+17.2 [14.6, 19.8]
Oracle	Qwen3-235B	64.4|77.8|80.4|93.6	81.5|91.2|90.3|97.6	+4.0 [2.7, 5.4]	77.6|87.7|84.6|89.5	88.7|96.6|94.3|97.9	+8.4 [6.7, 10.3]	51.0|65.7|63.0|75.6	70.3|84.7|81.1|90.7	+15.1 [12.5, 17.6]
Oracle	GPT-5.6	53.1|82.0|73.8|95.2	67.5|94.5|82.5|99.0	+3.8 [2.5, 5.2]	64.0|87.9|74.6|88.9	73.7|98.5|84.7|99.5	+10.6 [8.7, 12.5]	54.8|82.2|73.5|89.9	70.2|95.2|86.1|98.4	+8.5 [6.8, 10.4]

G.2 Gains on Samples Matched between the Original and Released Benchmarks

As Appendix G.1 notes, the original columns of Tables 3, 13 and 14 hold samples that the released columns leave out. Tables 15 and 16 repeat Tables 13 and 14 with the original benchmarks scored only on the samples that the released benchmarks keep. Each original (matched) column then holds exactly the same samples as the released column next to it, and the released columns are those of Tables 13 and 14. Each interval in brackets is computed as in Appendix G.1, with the resamples drawn from these samples.

The gains hold without the removed samples. On the matched samples every method and Oracle setting still scores higher under all four metrics, and the 95% interval of every LLM-judge gain lies above zero. The smallest LLM-judge gain is 2.1 points on the repaired benchmarks and 2.4 on the verified benchmark, and no gain differs by more than 0.4 and 1.8 points respectively from its value in Tables 13 and 14.

The gains therefore hold on exactly the same samples, for every method on all three benchmarks, and Appendix G.3 shows that they come mainly from the repaired samples.

Table 15: All four metrics on the samples that the repaired benchmarks keep. Original (matched) scores these samples as published, and the repaired column is that of Table 13. Each cell reads EM | Cover-EM | F1 | LLM judge, and Δ is the change in LLM-judge accuracy from original (matched) to repaired, with its 95% confidence interval in brackets.

Method	Base LLM	HotpotQA original (matched) $n = 7{,}374$	HotpotQA repaired $n = 7{,}374$	HotpotQA Δ	2WikiMultiHopQA original (matched) $n = 12{,}500$	2WikiMultiHopQA repaired $n = 12{,}500$	2WikiMultiHopQA Δ	MuSiQue original (matched) $n = 2{,}381$	MuSiQue repaired $n = 2{,}381$	MuSiQue Δ
Retrieval structure										
LinearRAG	Qwen3-32B	25.7|50.3|40.7|62.2	34.3|60.5|46.9|64.9	+2.7 [2.1, 3.3]	16.9|48.6|32.6|47.0	21.5|54.1|37.6|52.2	+5.2 [4.7, 5.7]	10.9|23.9|20.1|35.2	19.3|35.6|29.7|44.4	+9.2 [7.6, 10.9]
GraphRAG	Qwen3-32B	49.3|58.1|63.7|76.4	63.4|70.2|72.9|80.5	+4.1 [3.5, 4.7]	50.2|56.6|58.1|60.1	57.6|63.0|65.0|66.0	+5.9 [5.4, 6.4]	23.7|30.4|34.2|44.1	38.7|45.9|49.2|56.2	+12.1 [10.4, 13.7]
HippoRAG 2	Qwen3-32B	57.9|68.3|73.0|86.2	74.7|82.6|83.5|90.6	+4.4 [3.8, 4.9]	46.8|52.8|53.3|55.9	55.0|60.0|61.0|62.6	+6.7 [6.2, 7.2]	30.2|40.4|42.8|55.6	51.0|60.2|61.7|70.0	+14.4 [12.7, 16.1]
HippoRAG 2	GPT-5.6	56.8|74.1|76.2|93.2	72.7|88.3|86.4|97.3	+4.1 [3.6, 4.6]	60.9|86.2|78.5|93.0	70.0|90.6|85.1|96.7	+3.7 [3.3, 4.0]	43.9|60.3|60.4|74.0	61.1|76.0|76.0|85.9	+11.9 [10.4, 13.5]
Workflow										
LogicRAG	Qwen3-32B	47.9|55.1|61.9|74.8	61.0|66.6|70.5|78.1	+3.3 [2.7, 3.8]	59.2|68.4|68.1|71.4	66.4|74.2|74.9|76.8	+5.4 [4.8, 5.9]	26.5|35.3|38.2|49.1	41.6|49.2|52.3|59.8	+10.7 [8.9, 12.5]
STRIDE	Qwen3-32B	49.3|60.0|62.8|75.8	62.4|71.4|71.6|79.4	+3.6 [3.0, 4.2]	56.9|69.8|65.7|69.4	64.6|75.6|72.7|75.1	+5.7 [5.2, 6.2]	28.5|40.2|39.2|51.8	43.4|53.7|53.3|61.8	+10.0 [8.1, 11.8]
IRCoT	Qwen3-32B	56.8|66.4|71.4|84.4	72.5|79.4|81.2|88.1	+3.7 [3.2, 4.3]	63.8|72.0|73.4|81.3	73.5|79.7|82.0|88.7	+7.4 [6.9, 7.9]	36.2|43.3|47.8|57.7	52.0|58.5|62.5|69.9	+12.2 [10.4, 13.9]
IRCoT	GPT-5.6	63.2|71.5|78.5|91.4	81.2|86.6|89.4|95.9	+4.5 [4.0, 5.0]	74.0|83.9|84.3|94.6	83.2|89.0|91.2|98.5	+3.9 [3.6, 4.3]	53.8|63.5|65.8|75.5	72.9|79.3|81.4|87.6	+12.1 [10.6, 13.6]
Agentic search										
Search-R1	SearchR1-7B	48.1|52.1|60.1|69.2	60.5|62.8|67.7|71.7	+2.5 [1.9, 3.1]	26.2|28.1|30.3|30.4	29.3|30.6|33.2|32.5	+2.1 [1.7, 2.7]	20.4|22.5|27.7|31.6	31.9|33.7|38.9|41.5	+9.9 [8.2, 11.6]
ReSearch	ReSearch-32B	57.1|61.8|70.8|83.2	73.1|75.9|80.9|87.5	+4.3 [3.7, 4.9]	78.0|81.2|82.3|85.3	86.3|88.5|89.9|91.6	+6.3 [5.8, 6.8]	38.4|42.2|47.8|55.4	55.7|58.8|63.8|69.1	+13.7 [12.0, 15.5]
Search-o1	Qwen3-32B	50.7|56.5|65.2|79.8	65.6|70.1|75.0|83.9	+4.1 [3.5, 4.7]	68.7|78.6|77.7|81.6	77.5|85.4|85.5|88.0	+6.4 [5.8, 6.8]	28.3|33.9|39.1|52.0	43.2|47.5|52.9|62.9	+10.9 [9.2, 12.7]
Search-o1	GPT-5.6	41.5|68.9|63.9|89.2	54.2|82.5|73.2|93.3	+4.1 [3.6, 4.6]	40.7|88.9|69.2|92.1	47.5|93.1|75.0|95.9	+3.8 [3.4, 4.2]	31.8|52.6|50.0|66.5	45.6|69.3|65.2|80.3	+13.8 [12.1, 15.5]
Oracle Settings										
Oracle	Qwen3-8B	54.0|73.1|71.7|91.3	70.6|87.3|82.0|95.0	+3.7 [3.3, 4.3]	64.3|79.8|74.5|82.2	76.8|90.3|85.9|91.9	+9.7 [9.1, 10.2]	37.3|57.3|50.3|68.8	60.1|76.9|71.7|84.6	+15.8 [14.2, 17.6]
Oracle	Qwen3-32B	59.5|74.9|76.6|92.4	76.4|89.6|87.0|96.6	+4.2 [3.8, 4.7]	71.6|85.4|80.7|87.2	84.1|96.4|92.1|97.5	+10.3 [9.7, 10.8]	46.2|61.8|57.8|70.5	66.5|80.8|76.8|86.5	+16.0 [14.4, 17.7]
Oracle	Qwen3-235B	62.2|74.3|78.3|93.4	78.8|88.6|88.2|96.9	+3.5 [3.1, 4.0]	75.4|86.4|82.9|88.7	87.1|96.1|93.3|97.6	+8.9 [8.4, 9.4]	51.8|65.7|63.4|76.2	70.6|83.5|80.6|90.0	+13.8 [12.2, 15.4]
Oracle	GPT-5.6	52.2|80.8|73.7|95.0	67.0|93.6|82.9|98.7	+3.7 [3.3, 4.2]	62.7|86.5|74.1|87.6	75.6|98.7|86.3|99.3	+11.7 [11.2, 12.3]	55.1|82.0|73.3|89.2	70.7|95.2|86.8|98.5	+9.3 [8.2, 10.6]

Table 16: All four metrics on the samples that the verified benchmark keeps. Original (matched) scores these samples as published, and the verified column is that of Table 14. Each cell reads EM | Cover-EM | F1 | LLM judge, and Δ is the change in LLM-judge accuracy from original (matched) to verified, with its 95% confidence interval in brackets.

Method	Base LLM	HotpotQA original (matched) $n = 920$	HotpotQA verified $n = 920$	HotpotQA Δ	2WikiMultiHopQA original (matched) $n = 924$	2WikiMultiHopQA verified $n = 924$	2WikiMultiHopQA Δ	MuSiQue original (matched) $n = 888$	MuSiQue verified $n = 888$	MuSiQue Δ
Retrieval structure										
LinearRAG	Qwen3-32B	27.1|52.9|42.4|63.3	35.4|62.7|48.6|66.5	+3.2 [1.7, 4.9]	17.1|49.5|32.3|47.3	20.2|55.4|35.9|52.3	+5.0 [3.5, 6.6]	11.4|25.3|21.2|36.1	20.8|38.5|32.1|46.6	+10.5 [7.7, 13.4]
GraphRAG	Qwen3-32B	53.6|61.7|66.4|78.0	66.2|72.9|75.0|81.6	+3.6 [2.1, 5.1]	50.8|55.6|57.5|58.9	58.2|62.8|64.6|65.8	+6.9 [5.2, 8.8]	25.1|31.6|35.4|45.9	40.3|47.6|50.5|57.2	+11.3 [8.3, 14.2]
HippoRAG 2	Qwen3-32B	61.2|71.4|75.3|86.5	76.4|84.0|84.8|91.0	+4.5 [3.0, 6.0]	49.2|55.1|55.6|58.0	56.7|62.4|63.0|65.0	+7.0 [5.3, 8.9]	30.5|41.7|43.8|56.8	51.4|61.3|62.8|70.5	+13.7 [10.9, 16.7]
HippoRAG 2	GPT-5.6	58.5|76.6|77.5|92.3	73.4|89.9|86.9|97.0	+4.7 [3.4, 6.2]	62.7|87.9|79.1|93.6	70.7|91.7|85.0|97.2	+3.6 [2.2, 5.0]	44.1|61.8|60.8|75.7	60.6|76.6|76.3|86.5	+10.8 [8.2, 13.4]
Workflow										
LogicRAG	Qwen3-32B	50.7|57.5|63.1|73.8	63.3|68.2|71.3|77.3	+3.5 [2.0, 5.0]	60.3|68.0|68.3|70.6	68.0|75.4|75.6|77.1	+6.5 [4.8, 8.4]	26.9|36.5|38.5|50.5	43.0|50.8|53.5|62.0	+11.5 [8.7, 14.6]
STRIDE	Qwen3-32B	53.8|64.0|65.6|76.8	65.4|73.8|73.6|80.4	+3.6 [2.0, 5.3]	58.1|71.2|66.8|72.0	65.6|77.2|73.1|76.7	+4.7 [3.0, 6.6]	30.0|42.2|40.7|54.8	45.4|56.4|55.4|63.9	+9.1 [6.2, 11.9]
IRCoT	Qwen3-32B	59.3|69.3|73.5|84.8	73.8|80.5|82.1|87.9	+3.1 [1.7, 4.6]	64.2|72.0|73.8|81.5	72.8|79.2|81.2|88.2	+6.7 [5.1, 8.4]	35.6|42.6|47.8|58.9	53.2|59.0|63.8|70.8	+11.9 [9.1, 14.8]
IRCoT	GPT-5.6	65.4|73.8|80.3|91.1	81.6|87.5|89.7|95.3	+4.2 [2.9, 5.7]	75.5|84.2|85.4|94.9	83.4|89.0|91.5|98.7	+3.8 [2.5, 5.2]	56.0|66.3|68.0|78.4	73.5|79.8|82.4|88.9	+10.5 [8.0, 12.8]
Agentic search										
Search-R1	SearchR1-7B	50.9|54.6|61.8|68.3	62.8|65.0|69.2|71.2	+2.9 [1.3, 4.7]	26.8|28.4|30.5|31.4	30.0|31.7|33.9|33.8	+2.4 [0.9, 4.0]	19.7|21.8|27.4|31.9	33.2|34.3|40.1|42.0	+10.1 [7.2, 13.0]
ReSearch	ReSearch-32B	59.8|64.8|72.6|83.0	74.8|78.2|82.3|88.3	+5.3 [3.6, 6.8]	77.9|81.1|82.4|85.3	85.7|87.4|89.2|91.0	+5.7 [4.1, 7.5]	39.2|43.5|48.2|55.7	56.8|59.7|64.1|69.0	+13.3 [10.5, 16.2]
Search-o1	Qwen3-32B	53.6|57.4|67.1|78.5	68.0|71.3|77.0|82.9	+4.4 [2.8, 6.1]	72.8|81.0|80.3|83.5	80.6|87.3|87.1|89.8	+6.3 [4.7, 8.0]	29.7|35.8|40.5|53.6	47.3|50.8|56.2|66.1	+12.5 [9.6, 15.5]
Search-o1	GPT-5.6	41.5|69.9|63.4|88.0	52.4|81.8|71.4|91.6	+3.6 [2.2, 5.0]	39.4|91.5|68.3|93.4	44.8|95.1|73.5|96.6	+3.2 [1.9, 4.5]	32.1|52.5|50.1|66.8	43.4|67.7|62.9|78.6	+11.8 [9.1, 14.6]
Oracle Settings										
Oracle	Qwen3-8B	58.2|77.2|74.8|92.4	72.8|88.7|83.7|95.5	+3.1 [1.8, 4.6]	66.9|80.7|75.8|82.6	77.7|90.0|85.7|91.5	+8.9 [6.9, 10.8]	38.0|58.7|50.9|69.6	60.6|77.5|72.0|84.9	+15.3 [12.4, 18.1]
Oracle	Qwen3-32B	62.3|78.6|78.8|93.3	77.8|91.7|87.9|97.1	+3.8 [2.6, 5.1]	74.5|86.5|82.6|88.1	85.4|96.4|92.8|97.6	+9.5 [7.7, 11.6]	45.6|62.7|57.9|70.9	66.9|82.4|77.6|87.8	+16.9 [14.3, 19.6]
Oracle	Qwen3-235B	66.5|79.1|81.6|94.1	81.5|91.2|90.3|97.6	+3.5 [2.2, 4.8]	78.1|87.8|84.9|89.7	88.7|96.6|94.3|97.9	+8.2 [6.4, 10.2]	52.1|67.2|63.8|76.5	70.3|84.7|81.1|90.7	+14.2 [11.7, 16.8]
Oracle	GPT-5.6	53.9|83.0|74.2|95.5	67.5|94.5|82.5|99.0	+3.5 [2.2, 4.8]	64.2|88.0|74.9|89.2	73.7|98.5|84.7|99.5	+10.3 [8.3, 12.3]	55.3|82.5|73.3|89.5	70.2|95.2|86.1|98.4	+8.9 [7.1, 10.9]

G.3 Where the Gains Come From

Table 3 shows that every method scores higher on the released benchmarks. This section shows which samples the gains come from. Tables 17 and 18 split each gain Δ in Table 3 into four parts, on the repaired benchmarks and on the verified benchmark respectively. In an *answer-repaired* sample the question is kept, and the method's original output was scored again against the repaired answer. In a *question-rewritten* sample the question is rewritten, and the method was run again on the new question. For the Oracle rows, a sample whose gold documents change also counts as question-rewritten, since the method was run again on it (Appendix F.3). In an *unchanged* sample both the question and the answer are kept, and the method's original output was scored again. A *removed* sample counts only toward the original score, and the removed part is the change in the original score when these samples are left out. The four parts sum to Δ up to rounding, and the first row of each table gives the share of samples of each type for the RAG methods. On the repaired benchmarks the type of a sample follows the verdict of the LLM Auditor, and on the verified benchmark it follows the judgment of the two annotators, so a sample they disagree on or both judge `unfixable` counts as removed, and a `fixable` one takes the repair of its first round (Section 4.2.2). The 11 samples of Figure 6 whose proposed repair changed no field (Appendix E.3) are therefore answer-repaired here.

The gains come mainly from the repaired samples, above all from the question-rewritten ones. For every method and benchmark in both tables, the question-rewritten part is the largest of the four, and on average it accounts for 88% of the gain on the repaired benchmarks and 80% on the verified benchmark. On MuSiQue, which has the most defects of the three benchmarks, the question-rewritten samples make up under half of the samples (48% and 44%) but account for the great majority of the gain, 99% on average on the repaired benchmark and 90% on the verified benchmark. The answer-repaired samples contribute at most 1.4 points. Since neither the question nor the method's output changes in these samples, this part of the gain comes entirely from correcting the official answers. The unchanged samples change the gain by at most 0.3 points and the removed samples by at most 1.8, in line with Appendix G.2. The repaired and the verified benchmarks show the same pattern, so the conclusions hold when every sample is cross-checked by two annotators.

The gains therefore come mainly from repairing the defective samples, mostly by rewriting their questions.

Table 17: The gain of each method from the original to the repaired benchmark, in points of LLM-judge accuracy on the full development sets, split by repair type. Δ is the gain reported in Table 3. The first row gives the share of samples of each type for the RAG methods.

		HotpotQA					**2WikiMultiHopQA**					**MuSiQue**				
Method	Base LLM	Δ	answer repaired	question rewritten	unchanged	removed	Δ	answer repaired	question rewritten	unchanged	removed	Δ	answer repaired	question rewritten	unchanged	removed
Share of samples (%)		100	19.4	16.0	64.1	0.4	100	6.5	19.7	73.2	0.6	100	7.2	48.4	42.9	1.5
Retrieval structure																
LinearRAG	Qwen3-32B	+2.7	+0.5	+2.2	-0.1	0.0	+5.2	+0.5	+4.7	+0.1	0.0	+9.1	+0.2	+9.0	0.0	-0.1
GraphRAG	Qwen3-32B	+4.1	+1.0	+3.1	0.0	0.0	+5.8	+0.5	+5.4	0.0	0.0	+11.8	+0.3	+11.8	0.0	-0.3
HippoRAG 2	Qwen3-32B	+4.3	+0.9	+3.4	0.0	0.0	+6.7	+0.6	+6.1	0.0	0.0	+14.2	+0.3	+14.1	0.0	-0.2
HippoRAG 2	GPT-5.6	+4.1	+1.1	+3.0	0.0	0.0	+3.7	+0.6	+3.0	0.0	0.0	+11.8	+0.8	+11.3	0.0	-0.1
Workflow																
LogicRAG	Qwen3-32B	+3.2	+0.9	+2.4	0.0	-0.1	+5.3	+0.6	+4.7	0.0	-0.1	+10.6	0.0	+10.5	+0.1	-0.1
STRIDE	Qwen3-32B	+3.6	+0.8	+2.8	-0.1	0.0	+5.7	+0.6	+5.0	+0.1	0.0	+9.8	0.0	+10.0	-0.1	-0.2
IRCoT	Qwen3-32B	+3.7	+1.0	+2.8	0.0	0.0	+7.4	+0.6	+6.8	0.0	0.0	+12.1	0.0	+12.3	0.0	-0.1
IRCoT	GPT-5.6	+4.5	+1.2	+3.3	0.0	0.0	+3.9	+0.9	+3.1	0.0	0.0	+12.0	+0.5	+11.5	0.0	-0.1
Agentic search																
Search-R1	SearchR1-7B	+2.6	+0.6	+1.9	0.0	0.0	+2.0	+0.3	+1.8	0.0	-0.1	+9.8	0.0	+10.0	0.0	-0.1
ReSearch	ReSearch-32B	+4.3	+0.9	+3.3	+0.1	0.0	+6.3	+0.8	+5.5	0.0	0.0	+13.4	+0.3	+13.3	+0.1	-0.2
Search-o1	Qwen3-32B	+4.1	+1.0	+3.1	-0.1	0.0	+6.3	+0.8	+5.6	0.0	0.0	+10.7	+0.1	+10.7	+0.2	-0.3
Search-o1	GPT-5.6	+4.0	+0.8	+3.3	0.0	0.0	+3.8	+0.7	+3.1	0.0	0.0	+13.5	+1.0	+12.9	-0.1	-0.3
Oracle Settings																
Oracle	Qwen3-8B	+3.7	+0.9	+2.9	-0.1	0.0	+9.6	+0.8	+8.8	0.0	-0.1	+15.6	+0.8	+15.1	0.0	-0.2
Oracle	Qwen3-32B	+4.2	+0.9	+3.3	0.0	0.0	+10.2	+0.8	+9.5	0.0	-0.1	+15.8	+0.2	+16.0	-0.1	-0.3
Oracle	Qwen3-235B	+3.5	+0.9	+2.6	0.0	0.0	+8.9	+0.8	+8.1	0.0	-0.1	+13.5	+0.5	+13.4	0.0	-0.3
Oracle	GPT-5.6	+4.1	+0.9	+2.9	0.0	+0.3	+11.6	+0.9	+10.9	0.0	0.0	+9.2	+1.0	+8.4	0.0	-0.1

Table 18: The gain of each method from the original to the verified benchmark, in points of LLM-judge accuracy on the 1,000 samples per benchmark drawn for human verification (Appendix E.1), split by repair type. Δ is the gain reported in Table 3. The first row gives the share of samples of each type for the RAG methods.

		HotpotQA					**2WikiMultiHopQA**					**MuSiQue**				
Method	Base LLM	Δ	answer repaired	question rewritten	unchanged	removed	Δ	answer repaired	question rewritten	unchanged	removed	Δ	answer repaired	question rewritten	unchanged	removed
Share of samples (%)		100	17.5	13.6	60.9	8.0	100	7.9	14.4	70.1	7.6	100	8.7	43.5	36.6	11.2
Retrieval structure																
LinearRAG	Qwen3-32B	+4.2	+0.3	+3.3	-0.3	+1.0	+4.7	+0.5	+4.3	+0.1	-0.3	+11.9	0.0	+10.4	+0.1	+1.4
GraphRAG	Qwen3-32B	+5.1	+0.9	+2.7	0.0	+1.5	+6.8	+1.0	+5.8	+0.1	-0.1	+12.9	+0.2	+11.3	-0.2	+1.6
HippoRAG 2	Qwen3-32B	+5.6	+0.5	+3.7	+0.2	+1.1	+7.8	+0.9	+6.4	-0.2	+0.8	+15.0	+0.1	+13.4	+0.2	+1.3
HippoRAG 2	GPT-5.6	+5.4	+1.2	+3.5	0.0	+0.7	+4.2	+0.8	+2.8	0.0	+0.6	+11.9	+0.9	+10.2	-0.3	+1.1
Workflow																
LogicRAG	Qwen3-32B	+4.2	+0.9	+2.7	-0.1	+0.7	+6.6	+0.9	+5.5	+0.1	+0.1	+11.7	+0.1	+11.6	-0.1	+0.2
STRIDE	Qwen3-32B	+4.7	+0.1	+3.4	+0.1	+1.1	+5.4	+0.4	+4.2	+0.1	+0.7	+10.9	-0.3	+9.5	-0.1	+1.8
IRCoT	Qwen3-32B	+4.2	+0.7	+2.5	0.0	+1.1	+7.0	+0.9	+5.7	+0.1	+0.3	+13.0	-0.3	+12.4	-0.1	+1.1
IRCoT	GPT-5.6	+5.0	+0.8	+3.4	+0.1	+0.8	+4.1	+1.0	+2.8	0.0	+0.3	+11.8	+0.2	+10.2	0.0	+1.3
Agentic search																
Search-R1	SearchR1-7B	+3.5	+0.8	+2.4	-0.2	+0.6	+2.3	+0.4	+2.2	-0.2	-0.1	+11.3	-0.5	+10.7	-0.1	+1.2
ReSearch	ReSearch-32B	+6.3	+0.9	+4.2	+0.1	+1.0	+6.5	+0.9	+5.0	-0.1	+0.8	+14.7	0.0	+13.3	0.0	+1.4
Search-o1	Qwen3-32B	+5.3	+0.8	+3.7	0.0	+0.9	+6.7	+0.6	+5.6	0.0	+0.4	+13.5	0.0	+12.2	+0.3	+1.0
Search-o1	GPT-5.6	+3.9	+0.4	+3.3	-0.1	+0.3	+3.5	+0.6	+2.7	-0.1	+0.3	+12.9	+1.0	+10.8	0.0	+1.1
Oracle Settings																
Oracle	Qwen3-8B	+3.4	+0.7	+2.5	0.0	+0.3	+9.7	+1.1	+7.8	0.0	+0.8	+16.3	+1.1	+14.3	-0.1	+1.0
Oracle	Qwen3-32B	+4.4	+0.5	+3.3	0.0	+0.6	+10.1	+1.1	+8.4	0.0	+0.6	+17.2	+0.2	+16.7	0.0	+0.3
Oracle	Qwen3-235B	+4.0	+0.7	+2.7	+0.1	+0.5	+8.4	+0.9	+7.4	0.0	+0.2	+15.1	+0.6	+13.5	+0.1	+0.9
Oracle	GPT-5.6	+3.8	+0.7	+2.9	-0.1	+0.3	+10.6	+1.2	+9.1	0.0	+0.3	+8.5	+1.4	+7.5	0.0	-0.4

G.4 Do the Repairs Make the Questions Easier?

Since the gains come mainly from the question-rewritten samples (Appendix G.3), this section tests whether the rewrites simply make the questions easier. We take the unchanged samples as the reference for normal questions, since they have no detected defect and are identical in the original and the released benchmarks. Tables 19 and 20 report, for each method on the repaired benchmarks and on the verified benchmark respectively, the accuracy on the question-rewritten samples before and after the repair and the accuracy on the unchanged samples. The before and after columns cover the same question-rewritten samples, and the answer-repaired and removed samples are not part of this comparison. We also checked whether the rewrites put the answer into the question, and counted the reasoning hops that the first-round annotators recorded before and after each repair (Table 21).

The repairs mainly restore the points that the defects took away, rather than making the questions easier. Before the repair, the accuracy on the question-rewritten samples is below the reference for every RAG method and Oracle setting in both tables, by about 20 points on average for the RAG methods. After the repair, it returns to about the reference on HotpotQA and 2WikiMultiHopQA, within 2 points on average for the RAG methods. For example, in the Oracle setting Qwen3-235B rises from 81.3 to 97.2 on the question-rewritten samples of the repaired HotpotQA, while the unchanged samples stand at 96.7.

The closed-book models see only the question, so they would be the first to benefit if the rewrites made the questions easier. Before the repair they mostly fall below the reference on the question-rewritten samples, as GPT-5.6 does on HotpotQA with 67.3 against 87.6. After the repair their accuracy on these samples is, on average, close to or below that on the unchanged samples, and on all three repaired benchmarks Qwen3-235B and GPT-5.6 score slightly lower on the rewritten questions than on the unchanged ones.

On MuSiQue, with the documents given, the accuracy on the rewritten questions ends a few points above the reference, by 4.5 points on average for the RAG methods on the repaired benchmark and by 5.3 on the verified benchmark. Without the documents, however, the closed-book models answer them about as well as the unchanged samples, within about one point on average, although these questions come from longer reasoning chains, with 25% four-hop questions against 9% among the unchanged samples. The extra points therefore come mainly from the repair itself, which rewrites these questions to ask what their gold documents support (Section 5.2), and not from easier questions.

The rewrites also keep the answer out of the question. In 99.6% of the 4,495 rewritten questions whose answer is not yes or no, the rewrite does not add the answer to the question, and the same holds for 678 of the 683 on the verified benchmark. The repairs also keep the reasoning depth of the original questions. Of the 1,056 `fixable` samples of the verified benchmark, 980 (92.8%) keep the same number of hops after the repair and only 66 (6.2%) have fewer.

The gains therefore come mainly from repairing defects that the samples already had, not from making the questions easier.

Table 19: LLM-judge accuracy on the full development sets, original to repaired benchmark. For each method, the accuracy on the samples whose question was rewritten, before and after the repair, and on the unchanged samples.

		HotpotQA			**2WikiMultiHopQA**			**MuSiQue**		
		question rewritten			question rewritten			question rewritten		
Method	Base LLM	before	after	unchanged	before	after	unchanged	before	after	unchanged
Retrieval structure										
LinearRAG	Qwen3-32B	51.6	65.4	63.8	32.2	55.8	51.1	26.7	45.0	41.8
GraphRAG	Qwen3-32B	60.4	79.5	79.5	38.4	65.9	66.4	33.1	57.0	54.0
HippoRAG 2	Qwen3-32B	71.1	92.2	89.9	39.5	70.2	60.6	44.7	73.4	65.3
HippoRAG 2	GPT-5.6	78.6	97.3	97.2	79.4	94.5	97.3	65.3	88.2	83.5
Workflow										
LogicRAG	Qwen3-32B	60.3	75.4	78.3	48.7	72.6	77.9	40.4	61.9	56.4
STRIDE	Qwen3-32B	60.8	78.5	80.3	46.9	72.2	76.2	42.8	63.2	59.7
IRCoT	Qwen3-32B	70.0	87.1	88.4	53.9	88.0	88.9	46.7	71.6	67.3
IRCoT	GPT-5.6	76.3	97.0	95.6	82.2	97.8	98.7	66.2	89.7	85.4
Agentic search										
Search-R1	SearchR1-7B	54.8	66.7	72.5	23.9	33.1	32.0	21.8	42.1	39.6
ReSearch	ReSearch-32B	65.0	85.5	87.8	60.7	88.6	92.6	44.2	71.2	66.1
Search-o1	Qwen3-32B	63.7	83.0	83.1	57.4	85.4	88.6	42.4	64.1	61.7
Search-o1	GPT-5.6	74.5	94.9	92.7	81.3	97.0	95.6	57.3	83.6	76.0
Oracle Settings										
Oracle	Qwen3-8B	77.4	95.1	94.7	50.3	94.3	91.4	58.5	88.5	79.8
Oracle	Qwen3-32B	76.3	96.4	96.5	50.5	97.8	97.5	58.2	90.1	83.3
Oracle	Qwen3-235B	81.3	97.2	96.7	57.1	97.6	97.6	65.8	92.4	87.5
Oracle	GPT-5.6	82.1	99.3	98.4	44.7	98.9	99.4	82.5	99.2	97.6
Closed-book: the question alone										
Closed-book	Qwen3-8B	31.8	38.8	34.8	22.0	26.0	32.9	15.7	18.5	17.3
Closed-book	Qwen3-235B	44.8	51.9	55.2	37.6	43.3	46.8	27.4	33.6	35.5
Closed-book	GPT-5.6	67.3	85.9	87.6	76.8	88.3	91.4	53.8	69.5	70.7

Table 20: LLM-judge accuracy on the 1,000 samples per benchmark drawn for human verification (Appendix E.1), original to verified benchmark. Same columns as Table 19.

		HotpotQA			**2WikiMultiHopQA**			**MuSiQue**		
		question rewritten			question rewritten			question rewritten		
Method	Base LLM	before	after	unchanged	before	after	unchanged	before	after	unchanged
Retrieval structure										
LinearRAG	Qwen3-32B	49.3	71.3	64.5	36.1	63.9	50.9	27.8	49.0	41.3
GraphRAG	Qwen3-32B	59.6	77.9	80.6	34.0	71.5	65.0	35.9	58.9	52.7
HippoRAG 2	Qwen3-32B	68.4	93.4	90.5	35.4	76.4	62.8	46.4	73.8	65.6
HippoRAG 2	GPT-5.6	75.0	98.5	96.4	76.4	94.4	97.7	69.0	89.9	83.3
Workflow										
LogicRAG	Qwen3-32B	59.6	77.9	75.7	36.1	71.5	78.3	41.1	64.8	58.2
STRIDE	Qwen3-32B	55.1	77.9	82.1	47.2	74.3	77.5	47.1	66.4	60.1
IRCoT	Qwen3-32B	72.8	89.7	87.7	52.8	89.6	88.2	48.5	73.8	66.7
IRCoT	GPT-5.6	75.0	97.8	95.2	79.2	97.2	99.0	70.1	91.0	88.5
Agentic search										
Search-R1	SearchR1-7B	51.5	67.6	71.6	22.2	36.1	33.2	22.3	44.1	38.8
ReSearch	ReSearch-32B	60.3	89.0	88.0	58.3	90.3	91.7	43.7	70.8	66.9
Search-o1	Qwen3-32B	62.5	87.5	79.3	54.2	90.3	90.7	41.8	66.7	66.4
Search-o1	GPT-5.6	72.1	94.1	91.5	79.2	96.5	96.4	57.9	80.0	76.8
Oracle Settings										
Oracle	Qwen3-8B	78.5	94.4	95.4	44.1	93.8	91.4	59.3	88.0	81.1
Oracle	Qwen3-32B	76.4	97.2	96.7	44.8	98.6	97.4	58.8	92.3	83.9
Oracle	Qwen3-235B	79.9	97.2	97.5	51.7	98.6	97.9	66.3	93.4	87.4
Oracle	GPT-5.6	80.6	99.3	98.9	40.7	98.6	99.7	83.9	99.1	97.5
Closed-book: the question alone										
Closed-book	Qwen3-8B	33.8	41.2	32.5	22.9	28.5	34.7	15.6	18.6	15.6
Closed-book	Qwen3-235B	46.3	53.7	54.8	39.6	56.2	51.2	29.0	34.9	33.6
Closed-book	GPT-5.6	63.2	84.6	87.4	77.8	90.3	91.7	55.4	71.5	72.4

Table 21: Reasoning hops recorded in the first round of human verification before and after the repair, for the 1,056 `fixable` samples of the verified benchmark.

Benchmark	Fixable samples	Fewer hops	Same hops	More hops
HotpotQA	311	10	299	2
2WikiMultiHopQA	223	9	212	2
MuSiQue	522	47	469	6
Pooled	1,056	66	980	10