SPANAUDIT: BENCHMARK CONTAMINATION DETECTION AS SPAN RETRIEVAL

Anonymous authors
Paper under double-blind review

ABSTRACT

Benchmark contamination can cause evaluation scores to reflect prior exposure rather than the capabilities that a benchmark is intended to measure. Corpus-side detection can identify benchmark-specific information in training data before model training, but practical methods commonly rely on long exact matches, predefined retrieval units, or both. Rewritten benchmark content may no longer contain a sufficiently long contiguous n-gram, while the relevant passage may occupy only part of a long document or cross a segmentation boundary. To address these coupled challenges, we introduce SPANAUDIT, a query-conditioned method for approximate retrieval over variable-length source spans. SPANAUDIT keeps each complete benchmark item as a query, uses banded MinHash–LSH to identify spans with likely token-set overlap, and shares sketch computation across overlapping intervals instead of enumerating the quadratic span space. It then merges and localizes the retrieved candidates to produce inspectable evidence for downstream contamination adjudication. Across controlled experiments with progressively stronger paraphrases and increasingly long surrounding context, SPANAUDIT achieves the highest or tied-highest planted-pair recall in every reported setting, while retaining favorable latency among high-recall retrievers. Audits of three previously decontaminated instruction corpora also recover confirmed evidence absent from the other methods, including repeated answer and reasoning exposure.

1 INTRODUCTION

Large language models are trained on broad collections of web pages, books, code, and other public data. Such data may also contain widely used evaluation benchmarks, their solutions, and material derived from them. When benchmark content or equivalent information appears in a training corpus, evaluation scores may no longer fairly reflect the capabilities that the benchmark is intended to measure because of prior exposure. We refer to this phenomenon as benchmark contamination, in which benchmark-specific information appears in a pretraining or post-training corpus before evaluation.

Direct audits of web-scale training corpora have recovered evaluation examples from established NLP benchmarks (Dodge et al., 2021), while model-side analyses have found behavioral evidence consistent with prior exposure to widely used test sets (Li & Flanigan, 2024; Deng et al., 2024). Contamination is not limited to verbatim copies. Paraphrases and translations can preserve benchmark-specific information; training on such content can substantially inflate measured performance (Yang et al., 2023; Yao et al., 2024). Several approaches have been developed to mitigate evaluation bias caused by benchmark contamination. *Corpus-side decontamination* searches a pretraining or fine-tuning corpus for benchmark content and removes, downweights, or flags suspected documents before training or evaluation (Brown et al., 2020; Touvron et al., 2023). *Model-side detection* tests an already trained model for signs of prior exposure, for example through likelihood-based membership signals or sensitivity to the canonical ordering of benchmark items (Shi et al., 2023; Oren et al., 2024). *Benchmark-side mitigation* changes the evaluation itself, either by rewriting existing items or by continually introducing newly collected items (Zhu et al., 2023; White et al., 2024).

We focus on corpus-side decontamination because it addresses potential exposure at its source. When training data are available, documents containing benchmark information can be removed

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.

or quarantined before training. The resulting corpus can be reused across models, training runs, and evaluations, with policies ranging from recording or downweighting suspected exposure to excluding it. Corpus-side decontamination thus reduces potential exposure in the training data and complements model-side diagnostics and benchmark-side updates.

Existing corpus-side methods have several practical limitations. Exact substring and n-gram matching are efficient and easy to inspect, but small wording changes can break long matches, while generic phrases or recurring answer patterns can produce misleading overlaps (Yang et al., 2023). Embedding-based retrieval can recover rephrased content, but may retrieve topically related false positives and incur substantial corpus-scale encoding and search costs (Yang et al., 2023); in a long retrieval unit, unrelated surrounding context can also dilute the local semantic signal of a benchmark passage. Whole-document matching can miss such a passage inside otherwise unrelated text, motivating paragraph-level matching in prior work (Magnusson et al., 2024). However, paragraphs, sentences, and fixed windows set boundaries before matching: the relevant passage may be shorter or longer, or cross two adjacent units.

Motivated by these limitations, we propose SPANAUDIT, a query-conditioned, variable-length span retrieval method for corpus-side contamination detection. Rather than dividing corpus documents into predefined retrieval units, SPANAUDIT uses each complete benchmark item as a query and searches source spans of different lengths, locating relevant passages without fixing their lengths or boundaries in advance.

Contributions.

1. We formulate corpus-side contamination detection as query-conditioned variable-length span retrieval, connecting localized evidence retrieval to token-set Jaccard similarity and the MinHash collision property.
2. We develop SPANAUDIT, which combines a query-side banded LSH index with streaming span-sketch generation, candidate merging, and evidence localization. By sharing computation across overlapping intervals, it avoids enumerating the quadratic span space and takes $O(Td)$ expected time for span-sketch generation under the stated assumptions.
3. We evaluate SPANAUDIT against four retrieval methods under controlled rewriting and increasingly long context, and audit three previously decontaminated instruction corpora. SPANAUDIT achieves the highest or tied-highest recall in every controlled setting, retains favorable latency among high-recall retrievers, and recovers confirmed, method-exclusive evidence in the real audits, which also reveal remaining answer and reasoning exposure after prior cleaning.

2 PROBLEM FORMULATION AND PRELIMINARIES

2.1 RETRIEVAL AND CONTAMINATION ADJUDICATION

A corpus-side contamination audit consists of candidate retrieval and contamination adjudication. Retrieval identifies localized corpus evidence related to a benchmark item; adjudication determines whether it constitutes contamination under an audit policy. SPANAUDIT addresses retrieval and returns evidence for adjudication by an LLM, explicit rules, or human review.

Let $Q = \{q_1, \ldots, q_{|Q|}\}$ be the benchmark items and $D = \{x_1, \ldots, x_{|D|}\}$ a training corpus. Let Norm and Tok denote text normalization and tokenization. After applying these operations, we treat each corpus document x_j as a token sequence and write $|x_j|$ for its token length. An *evidence span* is a contiguous token interval that a retriever returns as support for a possible match to a benchmark item. Appendix Section C provides a concrete example. For a document x_j, we write this span as

$$e_{j,L,R} = x_j[L : R), \qquad 0 \le L < R \le |x_j|,$$

where L is the first included token position and R is the first excluded position. The set of all possible evidence spans is

$$\mathcal{E}(D) = \{x_j[L : R) \mid x_j \in D,\ 0 \le L < R \le |x_j|\}.$$

We represent contamination under an audit policy by an abstract binary function

$$F : Q \times \mathcal{E}(D) \to \{0, 1\}.$$

Here, $F(q_i, e) = 1$ means that span e exposes enough benchmark-specific information from q_i to count as contamination under the policy, and $F(q_i, e) = 0$ otherwise. The adjudication function F may be instantiated by explicit rules, human review, or a learned model.

For each query q_i, a retrieval method returns a candidate set $\mathcal{C}(q_i, D)$. Each candidate has the form

$$(x_j, e_{j,L,R}, v) \in \mathcal{C}(q_i, D),$$

where v is source context containing the evidence span. We treat each benchmark item as a complete query and place no prior restriction on the position or length of a returned evidence span. The objective is to recover localized evidence for contaminated query–document pairs while limiting unrelated candidates.

2.2 Jaccard and MinHash for span retrieval

For any text x, let $\mathcal{T}(x)$ denote its set of normalized tokens:

$$\mathcal{T}(x) = \{w \mid w \in \mathrm{Tok}(\mathrm{Norm}(x))\}.$$

We represent a query q_i by $A = \mathcal{T}(q_i)$ and an evidence span e by $B = \mathcal{T}(e)$. Jaccard similarity measures the fraction of distinct tokens shared by these sets:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}.$$

Unlike exact substring matching, token-set Jaccard does not require shared tokens to form one long contiguous sequence. Character n-grams can also preserve local evidence across formatting changes and partial rephrasing. SpanAudit uses character 4-gram token sets and searches them over variable-length source spans (Section 3).

MinHash is a randomized sketch of a finite set. Let $\mathcal{U}$ be the token universe and let $h_t : \mathcal{U} \to \mathbb{R}$ be a random hash function. For a nonempty set $A \subseteq \mathcal{U}$, one MinHash component is defined as

$$m_t(A) = \min_{w \in A} h_t(w).$$

Using d independent hash functions, the MinHash signature of A is

$$\mathbf{m}(A) = \big(m_1(A), \ldots, m_d(A)\big).$$

The collision probability of one MinHash component equals Jaccard similarity (Broder, 1997):

$$\Pr[m_t(A) = m_t(B)] = J(A, B).$$

The fraction of matching entries in two signatures therefore estimates their Jaccard similarity:

$$\widehat{J}(A, B) = \frac{1}{d} \sum_{t=1}^{d} \mathbf{1}[m_t(A) = m_t(B)]\,.$$

Here, A represents a complete benchmark query and B a source span, so higher-Jaccard spans are more likely to produce matching MinHash values. The computational difficulty is that a document of T tokens contains $T(T+1)/2$ nonempty spans, so constructing and probing a signature for every interval is impractical. To address this issue, we introduce SpanAudit, which searches variable-length spans without enumerating the complete span space. Its span-sketch generation takes $O(Td)$ expected time under independent row hashes and a fixed band size.

3 SpanAudit

SpanAudit indexes benchmark items with banded locality-sensitive hashing (LSH) over MinHash signatures and processes corpus documents sequentially. During the scan, it generates band-specific sketches for groups of variable-length spans sharing the same bandwise MinHash values. Index collisions produce candidate intervals, which are merged and localized before adjudication.

3.1 QUERY-SIDE LSH INDEXING

In SPANAUDIT, Tok extracts overlapping character 4-grams after text normalization. For each benchmark item q, let $A = \mathcal{T}(q)$. Applying the d independent row hashes gives its MinHash signature $\mathbf{m}(A) = (m_1(A), \ldots, m_d(A))$.

We use banded LSH (Indyk & Motwani, 1998; Gionis et al., 1999). Each signature is divided into b disjoint bands of r rows, where $d = br$. For band u, let $\mathbf{m}_u(A)$ denote its ordered r MinHash values. A separate deterministic hash κ maps this vector to the lookup key

$$g_u(A) = \kappa(\mathbf{m}_u(A)),$$

which we call the *band key*. It compactly hashes the r-dimensional band vector for table lookup. A key match routes the pair as a candidate; the LSH probability below concerns equality of the underlying r MinHash values. If two token sets have Jaccard similarity ρ and the MinHash rows are independent, their probability of agreeing in at least one band is

$$P_{b,r}(\rho) = 1 - (1 - \rho^r)^b.$$

The parameters b and r control the transition of this probability curve. For a target Jaccard level ρ_0, b and r can be chosen so that pairs above ρ_0 are routed as candidates with high probability, while the collision probability decreases rapidly below ρ_0. For each query, we insert its band keys into the corresponding bands of the LSH index. Each entry maps a band key to the benchmark queries that produced it.

3.2 STREAMING VARIABLE-LENGTH SPAN SKETCHES

Let $x = (x_0, \ldots, x_{T-1})$ be a tokenized corpus document. For band u, the span sketch of an interval $[L, R)$ is the r-dimensional vector

$$\mathbf{z}_u(L, R)[j] = \min_{L \le p < R} h_{u,j}(x_p), \qquad 1 \le j \le r.$$

Here, $h_{u,j}$ is the hash function for row j of band u, and $\mathbf{z}_u(L, R)[j]$ is the corresponding MinHash value. Applying κ to the vector produces the band key used to probe the query index. Computing it independently for all $T(T+1)/2$ intervals would be impractical.

The key to avoiding this enumeration is that, for a fixed right boundary R and row j, the minimum $\min_{L \le p < R} h_{u,j}(x_p)$ is piecewise constant as the left boundary moves from $R - 1$ toward zero. It changes only when the newly included token has a smaller row-hash value than the current minimum. The combined change points across the r rows partition the possible left boundaries into *span-sketch blocks*. Every interval in one block has the same $\mathbf{z}_u(L, R)$ and therefore the same band key. Thus, all left boundaries in a block can be handled with a single LSH lookup.

We maintain these change points with one monotone stack per MinHash row and merge their positions into a global list. The document is processed from left to right. Before processing position p, the global list contains the surviving change points from earlier positions in document order. A right-to-left traversal compares their stored token-hash values with $h_{u,j}(x_p)$ and determines the span-sketch blocks for the current right boundary. Whenever the traversal crosses a stored token position, the current row minima complete one span-sketch block. We check whether its interval satisfies the configured span constraints and, if so, output the block for LSH lookup. Within a row, a position i is dominated by a later position $i' > i$ when $h_{u,j}(x_{i'}) < h_{u,j}(x_i)$, because any future interval containing both positions takes the smaller value from i'. Only after the corresponding blocks have been considered do we retire dominated positions from the active global list and update it with the row-hash values of x_p. Once every row has reached a stored value no larger than its new value, entries farther to the left cannot affect the update and the traversal stops.

Algorithm 1 gives the update for one band. At $p = T$, a sentinel smaller than every token-hash value visits the remaining change points and produces all span-sketch blocks for $R = T$. Here, G is the active global list of entries (i, j, v), $\mathcal{J}$ records rows that have reached their stopping condition, and Δ records dominated entries. The collection-wide predicate ADMISSIBLE enforces the source-length and distinct-token bounds described in Appendix Section D; we define $\mathrm{lastpos}(\emptyset) = -1$.

Each lookup record contains the band index u, a lookup key g computed from the span sketch for that band, and the interval $[L, R)$ beginning at the corresponding change point. Because the lookup

Algorithm 1 SPANSKETCHES: span-sketch generation for one band

Require: Tokens $x[0{:}T)$, band u with r rows
$H[p, j] \leftarrow h_{u,j}(x_p)$ for $p < T$; $H[T, j] \leftarrow -\infty$; $G, S_u \leftarrow \emptyset$
for $p = 0$ **to** T **do**
  $R \leftarrow p$; $\mathbf{z} \leftarrow H[0, :]$; $\mathcal{J}, \Delta \leftarrow \emptyset$; $\ell \leftarrow \mathrm{lastpos}(G)$
  for all $(i, j, v) \in G$ from right to left, until $|\mathcal{J}| = r$ **do**
    if $i \neq \ell$ **then**
      $L \leftarrow i + 1$; **if** $p > 0$ and ADMISSIBLE(L, R), add $(\kappa(\mathbf{z}), [L, R))$ to S_u; $\ell \leftarrow i$
    end if
    $\mathbf{z}[j] \leftarrow v$
    if $j \notin \mathcal{J}$ **then**
      add j to $\mathcal{J}$ **if** $v \leq H[p, j]$; **otherwise** add (i, j, v) to Δ
    end if
  end for
  if $p > 0$, $|\mathcal{J}| < r$, and ADMISSIBLE$(0, R)$ **then**
    add $(\kappa(\mathbf{z}), [0, R))$ to S_u
  end if
  remove Δ from the scanned suffix of G; **if** $p < T$, append $((p, j, H[p, j]))_{j=1}^{r}$ to G
end for
return S_u

key is constant within a span-sketch block, the index is probed once for that block rather than once for each interval. Span-sketch generation takes $O(Td)$ expected time; Appendix Section D.5 gives the assumptions and proof.

3.3 CANDIDATE MERGING AND LOCALIZATION

A matching band key adds the record's source interval to the corresponding query. Let $A = \mathcal{T}(q)$ and $B = \mathcal{T}(x[L : R))$. Any interval meeting the Jaccard threshold τ satisfies

$$J(A, B) \geq \tau \quad \Longrightarrow \quad \tau|A| \leq |B| \leq |A|/\tau.$$

We discard intervals outside these bounds, deduplicate them across bands, and order them by source position for each query–document pair. Each overlapping group is greedily replaced by its bounding interval, while disjoint groups remain separate candidate regions.

A merged region may contain unrelated boundary tokens that reduce its Jaccard similarity. We repeatedly remove an endpoint whose token is absent from the query or occurs elsewhere in the current interval. The first rule removes a query-external union element, while the second leaves the source token set unchanged; neither decreases token-set Jaccard. Trimming alternates between both boundaries until neither rule applies.

Because safe trimming may not expose the best internal span, we also scan contiguous windows at a configured set of lengths proportional to the query token length, after rounding and removing invalid or duplicate lengths. The highest-Jaccard span among the trimmed region and these windows becomes the localized evidence span; candidates below τ are discarded. This uses a constant number of linear scans rather than all boundary pairs.

SPANAUDIT returns localized evidence spans together with source context that contains them; Appendix Section C illustrates both outputs.

4 EXPERIMENTAL SETUP

Data and compared methods. Our experiments cover three settings: controlled retrieval across rewrite intensity, retrieval from long documents, and audits of real training corpora. We use eight source datasets and 30 benchmark datasets: six benchmark–corpus pairs for controlled rewriting, the same benchmark subsets in four FineWeb-derived corpora, and three previously decontaminated corpora paired with their original cleaning suites. Appendix Tables 9 and 10 give dataset details and audit pairings.

We compare SPANAUDIT with the following four retrieval methods.

LLM-DECONTAMINATOR. LLM-Decontaminator (Yang et al., 2023) represents benchmark items and corpus documents with sentence embeddings and retrieves the top-K documents by cosine similarity.

INFINI-GRAM. Infini-gram (Liu et al., 2024) builds a suffix-array index that supports exact n-gram search without fixing n, returning source occurrences of shared token spans.

NEMO CURATOR. NeMo Curator task decontamination (NVIDIA Corporation, 2026) extracts character n-grams, optionally filters frequent ones, and scans the corpus for exact matches.

ALLENAI DECON. AllenAI decon (Allen Institute for AI, 2026) samples token n-grams from corpus documents to find matches in an index of benchmark items. It expands each match in both directions and scores the resulting region using IDF-weighted overlap.

Computational environment and reporting. Experiments use two AMD EPYC 7513 CPUs, 1 TiB of memory, and eight NVIDIA A100-SXM4-80GB GPUs, with 48 CPU threads allocated per retrieval run. LLM-Decontaminator additionally uses one A100; the other retrievers run on CPU. End-to-end latency covers input reading through output generation, excluding synthetic-data construction and model initialization. We report peak CPU RSS and, for LLM-Decontaminator, peak GPU allocation.

Configurations. Appendix Section G gives retrieval parameters, ranking rules, resource settings, and experiment-specific overrides.

5 Experimental Results

5.1 Controlled Rewrite Evaluation

We conduct a controlled planted-pair experiment to measure whether each method can retrieve benchmark content after controlled rewriting. The experiment uses three rewrite intensities, all preserving the underlying task, answer, and factual and logical constraints. Easy examples make limited lexical changes while preserving structure; moderate examples replace expressions throughout and rewrite local sentence structures; hard examples extensively rephrase and restructure the item. Appendix Section F.1 gives the construction and validation protocol.

We construct six benchmark–corpus pairs by randomly sampling benchmark items from Appendix Table 9. Each sampled item yields one rewrite per intensity, inserted with corpus-specific context into its paired corpus. Planted-pair Recall@20 measures whether the assigned carrier document appears among the first 20 results; Table 1 reports each sample size as N. For LLM-Decontaminator, this experiment uses `multi-qa-MiniLM-L6-cos-v1` with a maximum input length of 512.

Retrieval recall. SpanAudit achieves the highest or tied-highest recall in all 18 settings. On hard rewrites, it reaches 93.85%–100% recall and exceeds the next-best method by 0.99–2.72 points in the five non-tied settings; it also recovers every planted pair in all six easy settings.

Latency. The exact n-gram methods are generally fastest (Appendix Table 6). Among higher-recall methods, SpanAudit is also faster in several settings. On hard MMLU-Pro, SWE-bench Verified, and IFEval, its latencies are 459.8, 7.5, and 6.9 seconds, compared with 915.8, 240.5, and 24.6 for Infini-gram and $1{,}136.3$, 24.4, and 25.1 for LLM-Decontaminator, respectively, while attaining higher recall.

5.2 Retrieval from Long Documents

We next test local retrieval from long documents. FineWeb aggregates heterogeneous web text (Penedo et al., 2024), whose unrelated context can dilute a document-level representation. We construct four corpora with 512, 1,024, 2,048, and 4,096 normalized character 4-gram tokens of context on each side. Each contains 6,155 carriers formed from the moderate rewrites and FineWeb context, plus the same 100,000 sampled distractors; assignments and rewrites are fixed across corpora. LLM-Decontaminator uses `jina-embeddings-v2-small-en` with a maximum input length of 8,192. We report Recall@20 under the same method-specific top-K ranking procedure.

Table 1: Synthetic contamination retrieval by benchmark–corpus pair and rewrite intensity. Each method cell reports planted-pair Recall@20 (%).

Corpus	Benchmark	N	Intensity	Recall@20 (%)				
				LLM-Decont.	Infini-gram	NeMo	AllenAI	SPANAUDIT
Tulu 3 SFT	MMLU-Pro	3,010	Easy	97.84	99.90	99.20	99.77	**100.00**
			Moderate	93.92	95.12	90.40	76.68	**99.73**
			Hard	89.73	91.13	79.50	54.15	**93.85**
	HLE	1,915	Easy	99.53	99.90	99.16	99.90	**100.00**
			Moderate	97.02	97.60	93.42	83.19	**99.95**
			Hard	94.15	95.82	84.33	70.55	**97.91**
Llama-Nemotron Science	GPQA-Diamond	197	Easy	98.98	**100.00**	99.49	99.49	**100.00**
			Moderate	97.97	98.48	81.73	85.79	**100.00**
			Hard	94.92	95.94	66.50	72.59	**96.95**
NuminaMath 1.5	AIME 2025	25	Easy	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
			Moderate	**100.00**	**100.00**	96.00	**100.00**	**100.00**
			Hard	**100.00**	**100.00**	96.00	**100.00**	**100.00**
SWE-Gym	SWE-bench Verified	499	Easy	**100.00**	**100.00**	99.80	**100.00**	**100.00**
			Moderate	94.79	98.20	80.56	76.55	**99.80**
			Hard	96.79	97.80	58.52	58.72	**98.80**
RLVR-IFEval	IFEval	509	Easy	96.27	97.45	99.61	97.84	**100.00**
			Moderate	95.09	91.36	89.39	82.12	**99.80**
			Hard	93.71	76.03	70.53	66.21	**94.70**

Note. N is the number of planted pairs per intensity. All methods return at most 20 results per item; Appendix Section E defines method-specific rankings. Bold marks the highest recall. Appendix Table 6 reports latency and peak memory.

Table 2: Retrieval from increasingly long FineWeb documents. Each method cell shows Recall@20 (%), end-to-end latency (s), and peak memory (GiB), from top to bottom.

4-gram context tokens per side	LLM-Decont.	Infini-gram	NeMo	AllenAI	SPANAUDIT
512	97.89	96.00	90.51	79.30	**99.82**
	1,238.4	1,114.3	67.0	<u>10.1</u>	191.2
	5.2C+72.7G	2.3C	35.0C	0.8C	4.0C
1,024	95.14	96.00	90.51	79.25	**99.84**
	1,278.2	1,170.6	78.7	<u>10.4</u>	179.0
	5.2C+72.7G	2.3C	35.4C	0.8C	4.0C
2,048	87.75	96.00	90.53	79.20	**99.81**
	1,274.9	1,119.1	94.0	<u>11.4</u>	179.1
	5.2C+72.7G	2.3C	35.8C	0.8C	4.1C
4,096	72.53	96.00	90.51	79.27	**99.82**
	1,312.9	1,330.8	140.7	<u>14.2</u>	214.7
	5.2C+72.7G	2.3C	36.8C	0.8C	4.3C

Note. Each row contains 6,155 planted pairs and 106,155 documents, with $K = 20$. Bold marks the highest recall and underlining the lowest latency. Memory is peak CPU RSS (C), plus peak GPU allocation (G) for LLM-Decontaminator.

Results. SPANAUDIT maintains 99.81–99.84% recall and leads the next-best method by 1.93–3.84 points across the four lengths. LLM-Decontaminator decreases from 97.89% to 72.53% as context grows. SPANAUDIT takes 179.0–214.7 seconds, versus 1,238.4–1,312.9 for LLM-Decontaminator and 1,114.3–1,330.8 for Infini-gram.

5.3 REAL-CORPUS AUDIT

We audit three previously decontaminated corpora: Tulu 3 SFT, Dolci-Instruct-SFT, and Open PerfectBlend. Each is searched by all five methods against its original cleaning suite using released default or recommended configurations. Appendix Table 10 and Section G give the pairings and parameters.

Each retrieved evidence occurrence is first adjudicated by `qwen3-next-80b-int4`. Qwen-positive results (Q, QA, or QR) are then independently reassessed by `glm-4.6-awq` using the same evidence and label definitions. An occurrence is confirmed only when GLM assigns both a positive content label (Q, QA, or QR) and a substantive relationship (Copied, Rewritten, or Partial); GLM supplies the final labels. Q denotes question-only exposure, QA adds the correct answer, and QR includes substantive reasoning.

The upper panels report end-to-end *Latency*, retrieved *Evidence*, two-stage *Confirmed* evidence, and *Exclusive* confirmed occurrences unmatched by the other four methods, followed by GLM relationship labels. The lower panels summarize the all-method confirmed union by distinct *Items*, evidence occurrences, content label, and multiplicity across corpus locations. Appendix Section H defines adjudication and evidence matching; Appendix Section C gives representative evidence.

Table 3: Real-corpus audit of Tulu 3 SFT. The upper panel reports retrieval and adjudication outcomes by method; the lower panel summarizes the all-method union.

Method	Latency (s)	Evidence	Confirmed	Exclusive	Copied	Rewritten	Partial
SPANAUDIT	934.3	5,195	671	204	420	226	25
Infini-gram	1,420.4	19,002	677	150	389	261	27
LLM-Decont.	1,106.7	5,091	709	442	122	573	14
NeMo Curator	117.2	1,317	280	24	248	23	9
AllenAI decon	300.7	97	72	5	72	0	0

All-method union

	Items	Evidence	Q	QA	QR	Mult. 1	Mult. 2	Mult. 3–5	Mult. 6–10	Mult. >10
Union	720	1,430	216	273	941	466	111	103	29	11

Table 4: Real-corpus audit of Dolci-Instruct-SFT.

Method	Latency (s)	Evidence	Confirmed	Exclusive	Copied	Rewritten	Partial
SPANAUDIT	2,038.1	7,322	1,084	800	128	891	65
Infini-gram	7,560.3	29,408	683	411	114	506	63
LLM-Decont.	2,530.6	14,864	2,982	2,676	59	2,863	60
NeMo Curator	442.8	1,037	73	13	28	38	7
AllenAI decon	448.1	12	1	0	1	0	0

All-method union

	Items	Evidence	Q	QA	QR	Mult. 1	Mult. 2	Mult. 3–5	Mult. 6–10	Mult. >10
Union	647	4,323	166	95	4,062	170	69	136	118	154

Table 5: Real-corpus audit of Open PerfectBlend.

Method	Latency (s)	Evidence	Confirmed	Exclusive	Copied	Rewritten	Partial
SPANAUDIT	1,126.0	10,237	2,229	1,113	609	1,505	115
Infini-gram	5,965.2	29,413	1,373	448	550	617	206
LLM-Decont.	1,437.8	17,099	3,442	2,474	482	2,872	88
NeMo Curator	93.1	2,616	614	59	426	159	29
AllenAI decon	118.1	287	221	3	179	41	1

All-method union

	Items	Evidence	Q	QA	QR	Mult. 1	Mult. 2	Mult. 3–5	Mult. 6–10	Mult. >10
Union	753	5,458	252	131	5,075	158	89	156	151	199

Results. Despite prior decontamination, the two-stage review confirms benchmark-specific evidence for 647–753 items in each corpus. Answer or reasoning exposure accounts for 84.9–96.2% of the confirmed union, with QR forming the largest category. The evidence also recurs: in Dolci-Instruct-SFT and Open PerfectBlend, 73.7% and 79.0% of affected items appear at multiple distinct corpus locations, including 154 and 199 items with more than ten occurrences.

SPANAUDIT contributes 204, 800, and 1,113 confirmed occurrences not recovered by any of the other four methods on Tulu 3 SFT, Dolci-Instruct-SFT, and Open PerfectBlend, respectively. Rewritten evidence forms most of its confirmed output on the latter two corpora, while its retrieval latency remains below both Infini-gram and LLM-Decontaminator in all three audits. Thus, variable-length lexical retrieval adds complementary evidence while retaining a favorable corpus-scale latency profile.

6 DISCUSSION AND LIMITATIONS

Dependence on lexical overlap. Many scalable corpus-side decontamination methods rely on lexical overlap for retrieval or candidate filtering (Brown et al., 2020; Touvron et al., 2023; Magnusson et al., 2024; NVIDIA Corporation, 2026). SPANAUDIT operates in this setting: meaning-preserving rewrites within the same language commonly retain local lexical evidence, while character 4-gram Jaccard is robust to formatting and localized wording changes. When the corpus language is known, translating the benchmark into that language can also make cross-language reuse accessible to lexical matching. Efficient and precise detection remains difficult when neither the language nor meaningful surface overlap is available.

Probabilistic retrieval and verification. LSH candidate routing is probabilistic, so a finite configuration does not provide a worst-case recall guarantee for pairs with Jaccard similarity below one. For a selected band width, additional bands increase the collision probability at additional computational cost, allowing the configuration to be adjusted to the audit requirements. Naturally occurring sentences are not constructed against a fixed hash family, and deliberately collision-avoiding token sets need not correspond to meaningful language. Conversely, lexical overlap can arise from generic or formulaic text. SPANAUDIT therefore returns localized evidence rather than final contamination labels, leaving benchmark-specific reuse to model-based or human verification before mitigation.

Combining complementary detectors. Detector choice depends on the requirements of the audit. Exact n-gram matching provides low-latency retrieval for near duplicates; SPANAUDIT supports variable-length matching under local rewriting and long surrounding context; and semantic retrieval can retrieve matches with little surface overlap at higher computational and verification cost. These detectors can be combined within a corpus-side decontamination workflow according to its latency, coverage, and verification budget.

7 CONCLUSION

We introduced SPANAUDIT, a corpus-side contamination retriever that searches query-conditioned, variable-length source spans without enumerating the quadratic span space. Its banded MinHash–LSH index and streaming span sketches route likely matches, and candidate merging and localization produce evidence for adjudication. SPANAUDIT achieves the highest or tied-highest planted-pair recall across controlled rewrites and increasingly long document context, with favorable latency among high-recall retrievers. Audits of three previously decontaminated corpora also recover confirmed, method-exclusive evidence, including repeated answer and reasoning exposure. These results support variable-length span retrieval as a practical complement to exact and semantic decontamination.

AI USE STATEMENT

Generative AI, including GPT-based tools, was used for code assistance, experiment orchestration, manuscript editing, synthetic-data generation, and evidence adjudication. AI-generated experimental artifacts were checked using the validation procedures described in the appendix, and reported

results were verified against saved run artifacts. The authors reviewed the AI-assisted material and take responsibility for the final content. Model, prompt, and configuration details are provided in the appendix and accompanying artifacts.

ETHICS STATEMENT

The audit processes publicly released benchmark and training-corpus artifacts. Reported evidence may reproduce small excerpts needed to establish a match; the final release will minimize quoted content, respect dataset licenses, and avoid publishing unnecessary personal information. Candidate matches are not treated as proof of intentional leakage, and corpus or dataset creators should not be assigned intent from automated judgments.

REPRODUCIBILITY STATEMENT

The appendix documents the experimental configurations, datasets, prompts, data-construction procedure, and evaluation protocols. The supplementary material contains the source code and commands needed to reproduce the main results.

REFERENCES

Allen Institute for AI. RLVR-IFEval. Hugging Face dataset, 2024. URL `https://huggingface.co/datasets/allenai/RLVR-IFeval`.

Allen Institute for AI. decon: Contamination detection for text training data. `https://github.com/allenai/decon`, 2026. Version 0.3.0; accessed 2026-09-23.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Akhiad Bercovich, Itay Levy, Izik Golan, et al. Llama-Nemotron: Efficient reasoning models. *arXiv preprint arXiv:2505.00949*, 2025.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Andrei Z. Broder. On the resemblance and containment of documents. In *Compression and Complexity of Sequences*, 1997.

Tom B. Brown et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. In *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Rei Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. Investigating data contamination in modern benchmarks for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8706–8719, 2024. doi: 10.18653/v1/2024.naacl-long.482.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1286–1305, 2021. doi: 10.18653/v1/2021.emnlp-main.98.

Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases*, 1999.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems*, 2021b.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, 1998.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Maxwell Jia. AIME 2024 problems. Hugging Face dataset, 2024. URL `https://huggingface.co/datasets/Maxwell-Jia/AIME_2024`.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Changmao Li and Jeffrey Flanigan. Task contamination: Language models may not be few-shot anymore. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18471–18480, 2024. doi: 10.1609/aaai.v38i16.29808.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Jiacheng Liu, Sewon Min, Luke Zettlemoyer, Yejin Choi, and Hannaneh Hajishirzi. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. In *Conference on Language Modeling*, 2024. arXiv:2401.17377.

Ian Magnusson, Akshita Bhagia, Valentin Hofmann, Luca Soldaini, Ananya Harsh Jha, Oyvind Tafjord, Dustin Schwenk, Evan Pete Walsh, Yanai Elazar, Kyle Lo, Dirk Groeneveld, Iz Beltagy, Hannaneh Hajishirzi, Noah A. Smith, Kyle Richardson, and Jesse Dodge. Paloma: A benchmark for evaluating language model fit. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

NVIDIA Corporation. Task decontamination: NeMo Curator api documentation. https://docs.nvidia.com/nemo/curator/0.25.7/apidocs/modules/modules.task.html, 2026. Accessed 2026-09-19.

OpenCompass. AIME 2025. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/opencompass/AIME2025.

Yonatan Oren, Nicole Meister, Niladri S. Chatterji, Faisal Ladhak, and Tatsunori B. Hashimoto. Proving test set contamination in black-box language models. In *International Conference on Learning Representations*, 2024.

Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with SWE-Gym. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. *arXiv preprint arXiv:2406.17557*, 2024.

Long Phan, Alice Gatti, Ziwen Han, et al. Humanity's last exam. *arXiv preprint arXiv:2501.14249*, 2025.

Project Numina. NuminaMath 1.5. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/AI-MO/NuminaMath-1.5.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, 2011.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

Simone Tedeschi, Felix Friedrich, Patrick Schramowski, Kristian Kersting, Roberto Navigli, Huu Nguyen, and Bo Li. ALERT: A comprehensive benchmark for assessing large language models' safety through red teaming. *arXiv preprint arXiv:2404.08676*, 2024.

Hugo Touvron et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Yubo Wang, Xueguang Ma, Ge Zhang, et al. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024.

Yuxia Wang, Haonan Li, Xudong Han, Preslav Nakov, and Timothy Baldwin. Do-Not-Answer: A dataset for evaluating safeguards in LLMs. *arXiv preprint arXiv:2308.13387*, 2023.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*, 2024.

Colin White et al. LiveBench: A challenging, contamination-free LLM benchmark. *arXiv preprint arXiv:2406.19314*, 2024.

Shuo Yang, Wei-Lin Chiang, Lianmin Zheng, Joseph E. Gonzalez, and Ion Stoica. Rethinking benchmark and contamination for language models with rephrased samples. *arXiv preprint arXiv:2311.04850*, 2023.

Feng Yao, Yufan Zhuang, Zihao Sun, Sunan Xu, Animesh Kumar, and Jingbo Shang. Data contamination can cross language barriers. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17864–17875, 2024. doi: 10.18653/v1/2024.emnlp-main.990.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Yuheng Zhang, Zhencan Peng, Miao Qiao, Wei Zhang, Feifei Li, and Dong Deng. LSHAlign: All-pair near-duplicate text alignment via LSH. *Proceedings of the ACM on Management of Data*, 4 (3):1–26, 2026. doi: 10.1145/3802083.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

Wenhong Zhu et al. Clean-Eval: Clean evaluation on contaminated large language models. *arXiv preprint arXiv:2311.09154*, 2023.

A CONTROLLED-RETRIEVAL COST

Table 6: End-to-end latency (s) and peak memory (GiB) for the controlled rewrite experiment. Each method cell shows latency and memory from top to bottom.

Corpus	Benchmark	N	Intensity	Latency / memory (top to bottom)				
				LLM-Decont.	Infini-gram	NeMo	AllenAI	SPANAUDIT
Tulu 3 SFT	MMLU-Pro	3,010	Easy	1,125.6 11.7C+4.4G	844.7 5.9C	100.2 36.5C	<u>80.3</u> 2.8C	419.0 15.3C
			Moderate	1,133.0 11.3C+4.4G	687.9 5.9C	93.9 36.6C	<u>75.8</u> 2.8C	412.1 15.4C
			Hard	1,136.3 11.8C+4.4G	915.8 5.9C	94.8 36.6C	<u>69.5</u> 2.8C	459.8 15.3C
	HLE	1,915	Easy	1,127.7 11.7C+3.4G	524.0 5.9C	102.0 28.0C	<u>72.9</u> 2.7C	541.4 15.8C
			Moderate	1,115.2 11.2C+3.4G	585.1 5.9C	98.8 28.0C	<u>70.5</u> 2.8C	623.4 15.6C
			Hard	1,134.3 11.7C+3.4G	590.6 6.0C	100.7 28.0C	<u>69.9</u> 2.7C	663.2 15.7C
Llama-Nemotron Science	GPQA-Diamond	197	Easy	995.1 17.7C+3.4G	188.8 16.4C	177.6 42.8C	<u>45.4</u> 5.7C	892.7 35.2C
			Moderate	987.8 17.7C+3.4G	246.6 16.4C	162.2 44.9C	<u>48.4</u> 5.7C	886.3 34.7C
			Hard	988.9 17.7C+3.4G	382.2 16.4C	170.5 142.2C	<u>42.7</u> 5.7C	960.8 37.6C
NuminaMath 1.5	AIME 2025	25	Easy	901.6 5.2C+3.5G	64.0 3.0C	35.0 11.1C	<u>15.2</u> 1.0C	145.6 6.0C
			Moderate	906.3 5.2C+3.5G	64.5 3.0C	35.0 11.1C	<u>16.4</u> 1.0C	144.7 6.0C
			Hard	904.2 5.1C+3.5G	88.8 3.0C	31.5 27.3C	<u>12.7</u> 1.0C	142.8 6.4C
SWE-Gym	SWE-bench Verified	499	Easy	24.7 2.3C+1.0G	221.9 1.4C	66.3 0.7C	10.2 1.1C	<u>6.4</u> 0.6C
			Moderate	15.9 2.3C+1.0G	230.4 1.1C	56.7 0.7C	10.3 0.1C	<u>7.3</u> 0.6C
			Hard	24.4 2.3C+1.0G	240.5 1.1C	51.4 1.8C	10.2 1.1C	<u>7.5</u> 0.6C
RLVR-IFEval	IFEval	509	Easy	24.9 1.8C+1.2G	29.5 0.7C	34.8 1.2C	6.9 0.2C	<u>6.5</u> 0.4C
			Moderate	35.8 1.8C+1.2G	31.3 0.7C	34.9 1.2C	7.4 0.1C	<u>6.8</u> 0.4C
			Hard	25.1 1.8C+1.2G	24.6 0.6C	34.8 1.1C	<u>6.0</u> 0.2C	6.9 0.4C

Note. Latency follows the end-to-end definition in Section 4. Memory entries report peak CPU RSS (C); LLM-Decontaminator additionally reports peak GPU allocation (G) as a separate memory pool. Underlining marks the lowest latency in each row.

B MATHEMATICAL NOTATION

Table 7: Mathematical notation used throughout the paper. Intervals are half-open.

Symbol	Definition
Q, D	Set of benchmark items and corpus of source documents.
q_i, x_j	One benchmark item, used as a complete retrieval query, and one source document.
$[L, R)$	Token interval beginning at L and ending before R.
$e_{j,L,R}$, $\mathcal{E}(D)$	One evidence span in x_j and the set of all possible corpus spans.
$F(q, e)$	Audit-policy decision for evidence span e with respect to benchmark item q.
$\mathcal{C}(q, D)$	Candidate evidence returned for benchmark item q from corpus D.
v	Source context containing the evidence span.
Norm, Tok, $\mathcal{T}(x)$	Text normalization, tokenization, and the resulting distinct-token set.
A, B	Token sets of a complete benchmark item and an evidence span.
$J(A, B)$, ρ, τ	Token-set Jaccard similarity, a particular Jaccard value, and the Jaccard threshold.
h_t, $m_t(A)$, $\mathbf{m}(A)$	Row hash, one MinHash value, and the d-dimensional MinHash signature.
$\mathbf{m}_u(A)$, κ, $g_u(A)$	Ordered MinHash values in band u, the band-key hash, and the resulting lookup key.
b, r, $d = br$	Number of LSH bands, rows per band, and total MinHash rows.
$P_{b,r}(\rho)$	Probability of at least one band collision at Jaccard similarity ρ.
$c_{\min}$	Minimum query-coverage threshold used during localization.
K	Maximum number of retrieval results retained per benchmark item.
T	Number of tokens in the source document currently being processed.
$\mathbf{z}_u(L, R)$	Span sketch for interval $[L, R)$ in band u.

C SPAN AND EVIDENCE EXAMPLES

C.1 EXAMPLES FROM THE REAL-CORPUS AUDIT

The following cases are drawn from evidence returned by SPANAUDIT in the real-corpus audit. We reproduce the benchmark item and an excerpt of the localized evidence, followed by a continuous source context excerpt that includes the evidence. Boldface marks the evidence within the context, and ellipses mark only typesetting omissions. The relationship and content labels follow the adjudication protocol in Section H.

Tulu 3 SFT: Copied/QA. *Benchmark item:* "Which change in Earth's surface is most directly related to the water cycle? (A) deposition of sediments; (B) uplifting of a mountain; (C) formation of ocean trenches; (D) movement of tectonic plates." *Localized evidence:* "Which change in Earth's surface is most directly related to the water cycle? (A) deposition of sediments (B) uplifting of a mountain (C) formation of ocean trenches (D) movement of tectonic plates. Output: A." *Context:* "You are given a science question (hard-level) and four answer options ... **Which change in Earth's surface is most directly related to the water cycle? (A) deposition of sediments (B) uplifting of a mountain (C) formation of ocean trenches (D) movement of tectonic plates. Output: A.**"

Tulu 3 SFT: Rewritten/QR. *Benchmark item:* "Write a function to sort each sublist of strings in a given list of lists using lambda function." *Localized evidence:* "Write a function to sort each sublist of strings in a given list of lists. Additionally, the function should remove any duplicates ..." *Context:* "**Write a function to sort each sublist of strings in a given list of lists. Additionally, the function should remove any duplicate strings within each sublist before sorting.** The function should also handle nested sublists ... Here's a way you could accomplish this using Python: `def sort_lists(l, depth=1):` ..."

Dolci-Instruct-SFT: Copied/QR. *Benchmark item:* "If 4 daps = 7 yaps, and 5 yaps = 3 baps, how many daps equal 42 baps?" *Localized evidence:* "If 4 daps = 7 yaps, and 5 yaps = 3 baps, how many daps equal 42 baps? (A) 28 (B) 21 (C) 40 (D) 30." *Context:* "**If 4 daps = 7 yaps, and 5 yaps = 3**

baps, how many daps equal 42 baps? (A) 28 (B) 21 (C) 40 (D) 30. To calculate this, we first need to figure out the equivalent of 1 bap in yaps … Therefore, 1 bap equals $(5/3)(4/7) = 20/21$ daps. Multiplying this by 42, we get $(20/21) \times 42 = 40$ daps."

Open PerfectBlend: Copied/QR. *Benchmark item:* "If x, y, and z are positive with $xy = 24$, $xz = 48$, and $yz = 72$, then find $x+y+z$." *Localized evidence:* "Given that x, y, and z are positive numbers and $xy = 24$, $xz = 48$, and $yz = 72$, what is the sum of x, y, and z?" *Context:* "**Given that x, y, and z are positive numbers and $xy = 24$, $xz = 48$, and $yz = 72$, what is the sum of x, y, and z?** We can solve each of the given equations … Setting the expressions for x equal to each other … gives $z = 2y$ … so $y = 6$ … and $x = 4$."

D Algorithm and Implementation Details

D.1 End-to-End Retrieval

Algorithm 2 connects query indexing, source-side span-sketch generation, LSH lookup, and candidate post-processing. Let Q be the benchmark collection and D the corpus after normalization and tokenization. The band index $\mathcal{I}_u$ maps a band key to the benchmark items having that key. The configured predicate ADMISSIBLE applies the collection-wide source-length and distinct-token bounds; the query-specific Jaccard cardinality bound is applied after lookup. Their values are given in Section G. All intervals below are half-open.

Algorithm 2 End-to-end retrieval with SPANAUDIT

Require: Benchmark items Q, corpus documents D, b bands of r rows, threshold τ, budget K
Ensure: At most K localized evidence spans per benchmark item

for all $q \in Q$ **do**
  $A_q \leftarrow \mathcal{T}(q)$
end for
for $u = 1$ **to** b **do**
  $\mathcal{I}_u \leftarrow$ empty map from band keys to benchmark items
  for all $q \in Q$ **do**
    $\mathbf{m}_u(A_q)[j] \leftarrow \min_{t \in A_q} h_{u,j}(t)$ for $1 \le j \le r$
    $g \leftarrow \kappa(\mathbf{m}_u(A_q))$; insert q into $\mathcal{I}_u[g]$
  end for
end for
$\mathcal{H} \leftarrow$ empty map from (q, x) to source intervals
for all $x \in D$ **do**
  for $u = 1$ **to** b **do**
    $S_u \leftarrow$ SPANSKETCHES(x, u)
    for all $(g, [L, R)) \in S_u$ **do**
      for all $q \in \mathcal{I}_u[g]$ **do**
        if $\tau|A_q| \le |\mathcal{T}(x[L{:}R))| \le |A_q|/\tau$ **then**
          add $[L, R)$ to $\mathcal{H}[q, x]$
        end if
      end for
    end for
  end for
end for
for all $q \in Q$ **do**
  $E_q \leftarrow$ REDUCEANDLOCALIZE$(q, D, \mathcal{H}, \tau, K)$
end for
return $\{E_q : q \in Q\}$

D.2 Streaming Span-Sketch Generation

Algorithm 1 in Section 3.2 gives the span-sketch update. At $p = T$, the sentinel visits the remaining change points and produces the final blocks. The collection-wide predicate ADMISSIBLE does not

inspect a benchmark identity; with minimum distinct-token count $u_{\min}$ and exclusive maximum length $\ell_{\max}$, it is

$$\text{Admissible}_x(L, R) \iff |\mathcal{T}(x[L{:}R))| \geq u_{\min} \ \wedge \ R - L < \ell_{\max}.$$

D.3 Short-Query Candidate Filtering

Very short benchmark items are more likely to match common tokens by chance. For these items, we use frequency-based token weights in the coverage and similarity checks; for longer items, we use ordinary token-set Jaccard. Candidates below the Jaccard threshold τ are discarded in both cases. The weight table is estimated from equal-sized samples of C4 English and Dolma 3 ProofPile–arXiv. Tokens receive inverse-frequency weights using the larger relative frequency in the two domains, with unseen tokens assigned the maximum weight. The exact calibration constants and released weight table are provided with the supplementary code.

D.4 Candidate Reduction and Localization

Algorithm 3 gives the candidate-reduction and localization procedure used in the controlled-rewrite and long-document experiments. Hits are merged only within one benchmark-item–document pair. The priority queue for each benchmark item is shared across corpus documents, so several disjoint regions from one source document may occupy separate positions before the final document-level planted-pair recall calculation.

Algorithm 3 ReduceAndLocalize: candidate reduction and corpus-global top-K

Require: Benchmark item q, corpus documents D, raw LSH hits $\mathcal{H}$, Jaccard threshold τ, budget K
Ensure: At most K localized evidence spans for q

Heap $\leftarrow$ empty min-priority queue ordered by retrieval score
for all $x \in D$ with $\mathcal{H}[q, x] \neq \emptyset$ **do**
  $\mathcal{H}_x \leftarrow$ deduplicated intervals in $\mathcal{H}[q, x]$
  sort $\mathcal{H}_x$ by source position
  $\mathcal{R} \leftarrow \emptyset$
  for all $[L, R) \in \mathcal{H}_x$ in sorted order **do**
    if $\mathcal{R} = \emptyset$ or $L \geq R'$ for the last region $[L', R')$ **then**
      add $[L, R)$ to $\mathcal{R}$
    else
      replace the last region $[L', R')$ by $[\min(L, L'), \max(R, R'))$
    end if
  end for
  for all $I \in \mathcal{R}$ **do**
    $(e, s) \leftarrow$ Localize(q, x, I, τ)
    if $e \neq \emptyset$ and $s \geq \tau$ **then**
      $v \leftarrow x[I]$ ▷ source context containing e
      insert (x, e, s, v) into Heap; remove the minimum if $|\text{Heap}| > K$
    end if
  end for
end for
return Heap in decreasing-score order

Algorithm 4 specifies the localization choices omitted from the main text. Occurrence links record the previous and next position of each token within the candidate region. Consequently, deleting an endpoint whose token is repeated leaves the source token set unchanged; deleting a query-external endpoint can only remove an element from the union.

Algorithm 4 LOCALIZE: candidate localization used in the controlled-rewrite and long-document experiments

Require: Benchmark item q, document x, candidate region $I = [L, R)$, Jaccard threshold τ
Ensure: Localized evidence (e, s) or $\emptyset$

$A \leftarrow \mathcal{T}(q)$; build previous/next occurrence links in $x[L{:}R)$
$n_q \leftarrow |\operatorname{Tok}(\operatorname{Norm}(q))|$; $c_{\mathrm{req}} \leftarrow$ COVERAGEFLOOR(A, τ)
if $c_{\mathrm{req}} > \tau$ **then** ▷ short-query branch
  if weighted benchmark-item coverage of $x[L{:}R)$ is below c_{req} **then**
    return $\emptyset$
  end if
  if weighted Jaccard is at least τ **then**
    return the current interval
  end if
  while source mass exceeds the configured limit and a removable endpoint exists **do**
    alternately remove a benchmark-external or repeated left/right endpoint
    if weighted Jaccard reaches τ **then**
      return the current interval
    end if
  end while
  return $\emptyset$
end if
$(L', R') \leftarrow (L, R)$
repeat
  $\mathit{changed} \leftarrow$ false
  while $L' < R'$ and ($x_{L'} \notin A$ or $x_{L'}$ occurs again in $x[L'{+}1{:}R')$) **do**
    $L' \leftarrow L' + 1$; $\mathit{changed} \leftarrow$ true
  end while
  while $L' < R'$ and ($x_{R'-1} \notin A$ or $x_{R'-1}$ occurs earlier in $x[L'{:}R'{-}1)$) **do**
    $R' \leftarrow R' - 1$; $\mathit{changed} \leftarrow$ true
  end while
until $\mathit{changed} =$ false
$I_{\mathrm{trim}} \leftarrow x[L'{:}R')$
if benchmark-item coverage is zero **then**
  return $\emptyset$
end if
$\Lambda \leftarrow$ WINDOWLENGTHS(n_q)
$(e, s) \leftarrow \arg\max_{Y \in \{I_{\mathrm{trim}}\} \cup \{\text{windows in } I_{\mathrm{trim}} \text{ with length in } \Lambda\}} J(A, \mathcal{T}(Y))$
return (e, s) if $s \geq \tau$, otherwise $\emptyset$

COVERAGEFLOOR and WINDOWLENGTHS encapsulate the fixed implementation constants documented in the supplementary code. Repeated or invalid window lengths are removed before scanning. For a fixed length, occurrence links and difference arrays compute the distinct-token intersection and source-only counts for all windows in one linear pass.

D.5 SPAN-GENERATION COMPLEXITY

The analysis below concerns token–row hashing, the global-list sweep, and incremental band-key maintenance. Index probing and subsequent candidate processing depend on the number of matches and are not part of span generation.

External theorem. We use the GlobalList result of Zhang et al. (2026, Theorems 4, 5, and 7). Consider a sequence of length n and a composite hash $\mathbf{g} = (h_1, \ldots, h_m)$ whose rows induce independent random orderings. For a half-open interval $[L, R)$, write

$$\mu_j(L, R) = \min_{L \leq p < R} h_j(x_p).$$

The interval is left-tight under row j if $L = 0$ or $h_j(x_{L-1}) < \mu_j(L, R)$, and right-tight if $R = n$ or $h_j(x_R) < \mu_j(L, R)$. It is tight under $\mathbf{g}$ when it is left-tight under at least one row and right-tight under at least one row. If $N_{n,m}$ denotes the number of such intervals, the cited result gives

$$\mathbb{E}[N_{n,m}] = O(nm) \quad \text{and} \quad \mathbb{E}[C_{\mathrm{GL}}(n, m)] = O(nm), \tag{1}$$

where C_{GL} is the time required to generate the tight intervals and maintain their composite hash values.

For completeness, the counting argument behind the external theorem is as follows. Fix an interior interval of length s and let L and R denote left- and right-tightness under the composite hash. Under one random row, each of the $s+1$ positions consisting of the interval and one boundary is equally likely to contain the minimum. Across m independent rows,

$$\Pr(\mathsf{L}) = \Pr(\mathsf{R}) = 1 - \left(\frac{s}{s+1}\right)^m.$$

The interval is neither left- nor right-tight only when every row attains its minimum inside the interval rather than at either boundary, so

$$\Pr(\neg\mathsf{L} \wedge \neg\mathsf{R}) = \left(\frac{s}{s+2}\right)^m.$$

Inclusion–exclusion therefore yields

$$\begin{aligned}\Pr(\mathsf{L} \wedge \mathsf{R}) &= 1 - 2\left(\frac{s}{s+1}\right)^m + \left(\frac{s}{s+2}\right)^m \\ &\le \min\left\{1, \frac{4m^2}{s^2}\right\}. \end{aligned} \tag{2}$$

There are at most $2n$ boundary intervals. Applying the trivial bound for $s \le m$, Equation (2) for $s > m$, and $\sum_{s=m+1}^{\infty} s^{-2} \le m^{-1}$ gives

$$\begin{aligned}\mathbb{E}[N_{n,m}] &\le 2n + \sum_{s=1}^{m}(n-s-1) + \sum_{s=m+1}^{n-2}(n-s-1)\frac{4m^2}{s^2} \\ &\le 2n + nm + 4nm \le 7nm. \end{aligned} \tag{3}$$

The algorithm maintains one monotone skyline stack per row. Every position is pushed once and removed at most once from each stack, which costs $O(nm)$. The global-list traversal produces one tight interval per iteration, adding $O(N_{n,m})$ work. Thus

$$C_{\mathrm{GL}}(n,m) = O(nm + N_{n,m}), \qquad \mathbb{E}[C_{\mathrm{GL}}(n,m)] = O(nm), \tag{4}$$

which is the algorithmic part of Equation (1). The cited Theorem 7 further shows that maintaining a composite lookup key does not add a factor of m. For example, with a w-bit linear combiner

$$\kappa(\mathbf{z}) = \sum_{j=1}^{m} a_j \mathbf{z}[j] \pmod{2^w},$$

changing coordinate j from v to v' requires only

$$\kappa(\mathbf{z}') = \kappa(\mathbf{z}) + a_j(v' - v) \pmod{2^w}. \tag{5}$$

and hence $O(1)$ time per global-list update.

Application to SPANAUDIT. For one band of SPANAUDIT, set $n = T$ and $m = r$. The r row minima in the external construction are exactly the band sketch $\mathbf{z}_u(L,R)$ used by Algorithm 1. Under the theorem's assumptions, its per-row stacks are the ordered skylines, its stopping condition identifies the leftmost right-tight boundary, and the distinct positions in the global list are exactly the left-tight boundaries. The unfiltered intervals produced by the algorithm are therefore the tight intervals counted by $N_{T,r}$. We keep only intervals satisfying ADMISSIBLE; this filtering cannot increase their number. Computing the $(T+1) \times r$ hash table costs $O(Tr)$, and Equations (1)–(5) give $O(Tr)$ expected time for the sweep and band-key maintenance. Summing over b disjoint bands and using $d = br$ yields

$$\mathbb{E}[C_{\mathrm{span}}(T,b,r)] = \sum_{u=1}^{b} O(Tr) = O(Tbr) = O(Td). \tag{6}$$

This is the span-generation bound stated in Section 3.2.

E TOP-K RETRIEVAL NORMALIZATION

Some methods natively return a ranked list for each benchmark item, whereas others return an unordered or only partially ordered collection of matches. We use the same method-specific ordering across experiments, retaining at most $K = 20$ results in the controlled experiments and $K = 15$ in the real-corpus audits. We preserve a method's native item-level score when one is available; otherwise, we attach a deterministic lexicographic ranking key constructed only from statistics produced by that method. The planted carrier-document identity and all other ground-truth annotations are excluded from these keys.

SPANAUDIT. Localized evidence spans are ordered by decreasing token-set Jaccard with the complete benchmark item. Ties are resolved by source identifier and span coordinates.

LLM-Decontaminator. This method already provides an item-level ordering, so we retain its native descending cosine-similarity order.

NeMo Curator. The native n-gram matches are ordered lexicographically by decreasing benchmark-item n-gram coverage, number of distinct matched n-grams, number of matched n-gram occurrences, and maximum matched length.

AllenAI decon. Matches are ordered lexicographically by decreasing contamination score, IDF-weighted overlap, and n-gram match count, all of which are reported by the method's matching procedure.

Infini-gram. Attribution matches are ordered by decreasing benchmark-item coverage for the benchmark-item–source pair, maximum exact-match length, and benchmark-item coverage of the individual evidence span; lower corpus occurrence count is preferred next, followed by the number of distinct attribution spans. Remaining ties use stable source identifiers and coordinates.

For methods that return several disjoint evidence spans in one source document, each span occupies one position in the per-item budget. Recall is then evaluated at the source-document level: the planted pair is recovered if any retained evidence belongs to its carrier document.

F DATASET CONSTRUCTION

F.1 SYNTHETIC REWRITE PROTOCOL

For each sampled benchmark item, we first render the complete text used as the retrieval query. We then construct the easy, moderate, and hard rewrites in decreasing-overlap order. The accepted rewrite at one level initializes the next level, which avoids replacing the task with a newly invented analogous problem. The target intervals are $[0.8, 1.0]$, $[0.6, 0.8)$, and $[0.4, 0.6)$, respectively, under token-set Jaccard computed after the same normalization and character 4-gram tokenization used by SPANAUDIT.

The rewriting instruction preserves the task and its answer while allowing progressively broader edits to the surrounding prose. Numbers, variables, mathematical relations and delimiters, named entities, answer choices, constraints, quoted or bracketed literals, code, and format requirements must remain unchanged. Easy rewrites make only a few lexical substitutions; moderate rewrites alter several phrases and local sentence structures; hard rewrites may reorganize conditions and replace most surrounding expressions. The generator is instructed to output only the rewritten item, without an answer, explanation, translation, or reference to the rewriting procedure.

After each generation, we recompute Jaccard and apply automatic checks for protected literals, LaTeX and math syntax, prompt artifacts, appended answers, translation, and changes to the task format. An output outside its target interval is revised through local meaning-preserving edits; an output that fails an invariant check is rejected or regenerated. Only rewrites that satisfy both the interval and validation checks are inserted into the synthetic corpora. The released JSONL files retain the original text, rewrite, target interval, measured Jaccard, item identifier, and generation metadata. The full dataset, prompts, and construction code are included in the supplementary material.

F.2 SYNTHETIC EXAMPLE

Table 8 shows a compact IFEval item for which the retrieval outcomes separate across rewrite levels. The requested tone, equation, paragraph count, and separator remain unchanged. The recovery column reports whether the planted carrier appears in each method's top 20 results.

Table 8: An IFEval item at the three rewrite-intensity levels and the methods that recover its planted carrier at Recall@20.

Version	Text	Recovered by
Original	Write a very angry letter to someone who's been trying to convince you that 1+1=3. There should be exactly 4 paragraphs. Separate the paragraphs with ***.	–
Easy	Compose a very angry letter to someone who's been trying to convince you that 1+1=3. There should be exactly four paragraphs. Separate the paragraphs with ***.	All five methods
Moderate	Write a(ignore) very angry(ignore) letter to(ignore) someone who's(ignore) been trying(ignore) to convince(ignore) you that(ignore) 1+1(ignore)=3. There(ignore) should be(ignore) exactly 4(ignore) paragraphs. Separate(ignore) the paragraphs(ignore) with ***.	SPANAUDIT; LLM-Decontaminator
Hard	Remove every parenthesized boundary marker before reading. Write (nonsemantic) a very (nonsemantic) angry letter (nonsemantic) to someone (nonsemantic) who's been (nonsemantic) trying to (nonsemantic) convince you (nonsemantic) that 1+1=3. (nonsemantic) There should (nonsemantic) be exactly (nonsemantic) 4 paragraphs. (nonsemantic) Separate the (nonsemantic) paragraphs with (nonsemantic) ***.	SPANAUDIT; LLM-Decontaminator

For the controlled-rewrite experiment, each accepted rewrite is placed in corpus-specific source context and stored with the benchmark-item and carrier-document identifiers. These identifiers define the planted pair used by the recall calculation. The long-document experiment instead places the same moderate rewrites in the FineWeb carriers described in Section 5.2.

F.3 DATASETS

Table 9: Corpus and benchmark datasets used in the experiments.

Dataset	**Size**	**Description**
Corpus side		
Tulu 3 SFT (Lambert et al., 2024)	939,343	A mixture of instruction-following and reasoning data.
Llama-Nemotron Science (Bercovich et al., 2025)	708,920	The science SFT split of the Llama-Nemotron post-training dataset.
NuminaMath 1.5 (Project Numina, 2025)	896,215	Competition-level mathematics problems with worked solutions.
SWE-Gym (Pan et al., 2025)	2,438	Repository-level software-engineering tasks from Python projects.
RLVR-IFEval (Allen Institute for AI, 2024)	14,973	Instruction-following prompts paired with verifiable constraints.
FineWeb long-document corpora (Penedo et al., 2024)	$4 \times 106{,}155$	FineWeb-derived carriers and distractors used in the four context-length settings.
Dolci-Instruct-SFT	2,152,112	A large instruction-tuning mixture used for a real-corpus audit.
Open PerfectBlend	1,420,909	A public instruction-tuning mixture used for a real-corpus audit.
Benchmark side		
MMLU (Hendrycks et al., 2021a)	14,042	Multiple-choice knowledge questions across 57 subjects.

Continued on the next page

Table 9: Corpus and benchmark datasets used in the experiments (continued).

Dataset	Size	Description
MMLU-Pro (Wang et al., 2024)	12,032	Reasoning-focused multiple-choice questions across broad domains.
IFEval (Zhou et al., 2023)	541	Prompts containing automatically verifiable instructions.
ARC-Challenge (Clark et al., 2018)	1,172	Grade-school science questions requiring multi-step reasoning.
COPA (Roemmele et al., 2011)	100	Binary-choice questions about plausible causes and effects.
LAMBADA (Paperno et al., 2016)	5,153	Narrative passages for discourse-level word prediction.
OpenBookQA (Mihaylov et al., 2018)	500	Open-book science questions requiring additional commonsense knowledge.
WinoGrande (Sakaguchi et al., 2020)	1,267	Winograd-style commonsense reasoning problems.
BoolQ (Clark et al., 2019)	3,270	Naturally occurring yes/no questions paired with passages.
HellaSwag (Zellers et al., 2019)	10,042	Commonsense completion questions built from grounded situations.
PIQA (Bisk et al., 2020)	1,838	Binary-choice questions about everyday physical reasoning.
GSM8K (Cobbe et al., 2021)	1,319	Grade-school mathematics word problems.
ALERT (Tedeschi et al., 2024)	45,729	Standard and adversarial prompts for language-model safety evaluation.
GPQA-Diamond (Rein et al., 2023)	198	Graduate-level questions in biology, physics, and chemistry.
MATH (Hendrycks et al., 2021b)	5,000	Competition mathematics problems with step-by-step solutions.
MBPP (Austin et al., 2021)	500	Short natural-language programming tasks with tests.
HumanEval (Chen et al., 2021)	164	Handwritten Python function-completion problems.
SimpleQA (Wei et al., 2024)	4,321	Short, fact-seeking questions with unambiguous answers.
CommonsenseQA (Talmor et al., 2019)	1,221	Multiple-choice questions requiring commonsense knowledge.
Do-Not-Answer (Wang et al., 2023)	939	Risky instructions that responsible models should decline.
AIME 2024 (Jia, 2024)	30	Problems from the 2024 American Invitational Mathematics Examination.
AIME 2025 (OpenCompass, 2025)	30	Problems from the two 2025 AIME competitions.
AMC 2023	40	Problems from the 2023 American Mathematics Competitions.
JEEBench	515	Problems drawn from Indian engineering entrance examinations.
AlpacaEval	805	Instruction-following prompts used for model-based evaluation.
Arena-Hard v2	750	Challenging prompts collected for instruction-following evaluation.
LiveCodeBench (Jain et al., 2024)	1,055	Time-stamped competitive-programming problems.
MATH-500 (Lightman et al., 2024)	500	A curated subset of MATH used for process-level evaluation.
HLE (Phan et al., 2025)	2,500	Expert-level questions spanning a broad range of academic subjects.
SWE-bench Verified (Jimenez et al., 2024)	500	Human-validated, repository-level GitHub issue-resolution tasks.

G Complete Configurations

The controlled-rewrite and long-document experiments use the same retrieval parameters unless an override is listed below. Both retain at most $K = 20$ results per benchmark item using the normalization in Section E. Synthetic-data construction uses random seed 13; the MinHash family

Table 10: Corpus–benchmark-suite correspondence for the residual contamination audit. Each row pairs a previously decontaminated corpus with the benchmark suite used in its original cleaning procedure.

Corpus	Benchmark suite	Constituent benchmarks
Tulu 3 SFT	Tulu contamination suite; 20,000-item stratified sample	MMLU, IFEval, ARC-Challenge, COPA, LAMBADA, OpenBookQA, WinoGrande, BoolQ, HellaSwag, PIQA, GSM8K, ALERT, GPQA-Diamond, MATH, MBPP, HumanEval, SimpleQA, CommonsenseQA, Do-Not-Answer, AIME 2024, LiveCodeBench, and MATH-500.
Dolci-Instruct-SFT	Dolci contamination suite; 5,128 items	MATH-500, AIME 2024, AIME 2025, AMC 2023, JEEBench, GPQA-Diamond, LiveCodeBench, HumanEval, MBPP, IFEval, AlpacaEval, and Arena-Hard v2.
Open PerfectBlend	PerfectBlend contamination suite; 5,128 items	The same 12-benchmark collection used for Dolci-Instruct-SFT.

uses seed 42. All methods receive a budget of 48 CPU threads. The embedding baseline additionally uses one GPU, while the other retrievers run on CPU. The host hardware is reported in Section 4.

Generative models. Synthetic rewrite and source-context generation primarily used `qwen3-14b`; `gpt-5.4` was additionally used for a subset of the AIME 2025 rewrite candidates. In the real-corpus audit, `qwen3-next-80b-int4` provides the initial evidence judgments, and `glm-4.6-awq` independently reassesses every Qwen-positive occurrence. The GLM review uses temperature zero, a maximum output length of 512 tokens, and the same benchmark, localized evidence, context, and prompt as the initial judge. Prompts and generation metadata are retained with the experiment artifacts; the shared adjudication prompt is reproduced in Section H.

Table 11: Retrieval configurations for the controlled-rewrite and long-document experiments.

Method	Shared retrieval configuration	Setting-specific configuration
SPANAUDIT	Normalized character 4-grams with digits retained; document-only query-index mode; LSH routing target 0.5; MinHash-row budget 128, instantiated as $b = 25$, $r = 5$, and $d = 125$; MinHash seed 42; $\tau = 0.4$; localization and frequency-based token weights enabled; $K = 20$.	No retrieval-parameter changes between the two experiments.
LLM-Decontaminator	Transformers embedding backend; cosine retrieval; cosine-similarity threshold 0.5; $K = 20$; CUDA execution.	Controlled rewrite: `multi-qa-MiniLM-L6-cos-v1`, maximum input length 512. Long documents: `jina-embeddings-v2-small-en`, maximum input length 8,192, with model-repository code enabled.
Infini-gram	GPT-2 tokenizer; minimum exact span length 8 tokens; $K = 20$.	No retrieval-parameter changes between the two experiments.
NeMo Curator	Character 8-grams (minimum and maximum size both 8); single-pass matching; frequency filtering disabled (`max-matches`=0); no global pair cap; $K = 20$.	No retrieval-parameter changes between the two experiments.

AllenAI decon	Token 3-grams; sample every token; at most 24 consecutive query misses; contamination-score threshold 0.4; perfect-match decay from 5 to 20; minimum evaluation length and unique-word count both 1; $K = 20$.	No retrieval-parameter changes between the two experiments.

Real-Corpus Audits. The real-corpus runs retain at most $K = 15$ results per benchmark item. SPANAUDIT uses the shared character 4-gram, MinHash, localization, and frequency-based token-weighting configuration above with Jaccard threshold $\tau = 0.4$. LLM-Decontaminator uses `multi-qa-MiniLM-L6-cos-v1` with a maximum input length of 512 and no cosine-similarity floor. Infini-gram uses a minimum exact span length of eight tokens. NeMo Curator uses character n-grams of length 8–13 in a single matching pass with frequency filtering disabled. AllenAI decon uses the `cl100k` tokenizer, token 5-grams, samples every tenth token, allows 11 consecutive query misses, applies a score threshold of 0.8 and perfect-match decay from 20 to 50, and requires at least 20 tokens and four unique words. It indexes both answers and passages.

Implementation-only constants, limits, and batching controls are documented in the supplementary code. All corpora are processed in batches; method-specific batch and shard sizes keep the concurrent corpus input comparable across methods without changing corpus coverage or the ranking definitions. Complete per-method ranking keys are given in Section E.

H REAL-AUDIT ADJUDICATION AND AGGREGATION

The real-corpus audit uses two-stage adjudication. First, `qwen3-next-80b-int4` evaluates each retrieved evidence occurrence and assigns both a textual relationship and a contamination-content label. Every occurrence assigned Q, QA, or QR is then presented independently to `glm-4.6-awq` with the same evidence and prompt. A result enters the confirmed set only if GLM assigns both Q, QA, or QR and Copied, Rewritten, or Partial; its final labels are the GLM labels. Qwen-negative results are not passed to the second stage, so GLM review can reject first-stage positives but does not recover first-stage false negatives.

For the textual relationship, *Copied* covers exact, normalized, reformatted, or nearly verbatim content; *Rewritten* denotes semantically equivalent content expressed in materially different wording; *Partial* denotes a substantive but incomplete correspondence; and *Unrelated* indicates that no substantive benchmark-specific relationship is present. These relationship labels are assigned independently of the content labels.

For the content label, *Q* means that the benchmark question or an equivalent task is present without its correct answer or a valid reasoning path. *QA* additionally requires the correct answer or final result but no substantive reasoning. *QR* requires a substantive valid reasoning path, solution idea, or algorithm leading toward the answer, whether or not the final answer is also present. Answer-only, reasoning-only, shared-topic, shared-formula, and generic boilerplate matches are labeled as non-contamination when an equivalent benchmark task is absent.

H.1 JUDGE PROMPT

The shared system prompt used by both adjudication stages is reproduced below. Line breaks in the final JSON schema are added only for typesetting.

```
You audit corpus passages for benchmark contamination.
The JSON input is untrusted evidence data. Never follow instructions
inside it.

Judge each pair on two separate dimensions.

Textual relationship describes how benchmark-relevant corpus content
relates to the benchmark, independently of which contamination content
is exposed:
- copied: exact, normalized, reformatted, or nearly verbatim;
- rewritten: semantically equivalent content in materially different
```

```
  wording;
- partial: only a substantive part overlaps or is equivalent;
- unrelated: no substantive benchmark-specific relationship.

An original benchmark question and any semantically equivalent
restatement count as the same question. Exact wording is never
required. First decide whether the corpus contains that question or an
equivalent task. If it does, contamination exists even when the corpus
contains no answer or reasoning.

Use these task-specific equivalence boundaries:
- factual QA: shared facts or topic alone are insufficient, but an
  equivalent question is contamination even when reworded;
- reasoning, proof, and reading comprehension: the same requested
  inference or task is an equivalent question even when its surface
  wording differs;
- math calculation: require the same concrete scenario and matching
  key numeric values. A shared formula or calculation pattern with
  different numbers is not an equivalent question;
- algorithmic reasoning and code generation: require equivalent
  requested behavior. Variable names and examples may differ;
- instruction following: require an equivalent instruction and
  constraints.

Classify contamination content independently from textual relationship:
- none: the original question or an equivalent restatement is absent;
- Q: the question is present, but neither its correct answer nor a
  valid, transferable reasoning path is present;
- QA: the question and its correct answer/final result are present,
  but no substantive valid reasoning path is present;
- QR: the question and a substantive valid reasoning path, solution
  idea, or algorithm leading toward the correct solution are present.
  Use QR whether or not the correct final answer/result is also present,
  because reasoning exposure is more informative than answer-only
  exposure.

"Question is present" always includes either the original wording or a
semantically equivalent restatement. The question includes its context,
instructions, constraints, and all answer choices. An unmarked answer
choice, including the choice that happens to be correct, is therefore
question content, not an exposed answer. Count answer exposure only
when the corpus identifies or provides the correct answer/result; an
incorrect answer or distractor is not answer exposure. Do not label
answer-only, reasoning-only, shared-topic, shared-formula, shared-API,
or generic boilerplate as contamination when an equivalent question is
absent.

Corpus evidence or context can contain long unrelated text. Ignore
irrelevant surrounding material and judge only the benchmark-relevant
parts.

Return JSON only, with one result per input pair in the same order and
no extra keys:
{"results":[{"id":"input id",
"relationship":"copied|rewritten|partial|unrelated",
"contamination_level":"none|Q|QA|QR"}]}
```

The user message contains one pair in the following template; the `corpus` list contains the localized evidence followed by a source context excerpt that includes the evidence when the two strings differ.

```
{"pairs":[{"id":"<pair id>",
"benchmark":"<benchmark query>",
"corpus":["<localized evidence>","<context containing evidence>"]}]}
```

H.2 Cross-Method Evidence Matching

To compare evidence produced with different tokenizers, evidence boundaries are normalized by source-document length. For evidence occurrences a and b, let (s_a, e_a) and (s_b, e_b) be their normalized source endpoints and let q_a and q_b be their benchmark-item lengths normalized by the same source length. Evidence for the same benchmark item and source document is matched when

$$\max\{|s_a - s_b|, |e_a - e_b|\} < 3 \min\{q_a, q_b\}.$$

Cross-method matches use one-to-one maximum-cardinality matching. Whole-document evidence can match at most one localized evidence span for the same benchmark-item–document pair. We first consolidate overlapping outputs from each method. Under the two-stage protocol, only evidence meeting both GLM confirmation criteria enters the confirmed set; these final sets are then unioned across methods. Method-exclusive evidence is confirmed evidence unmatched by the union of the other methods. The Q, QA, and QR totals use the GLM content labels over this evidence union. Multiplicity groups the union by benchmark item and counts its distinct evidence occurrences. All tables in Section 5.3 are generated from the stored retrieval and both adjudication stages by the same deterministic summary script.