TRACEATTENTION: EFFICIENT LONG-CONTEXT ATTRIBUTION IN REASONING LLMS VIA CACHE-PRESERVING EVALUATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Context attribution explains how input spans influence model responses using internal-state analysis or perturbations. Perturbation-based methods score a fixed response under block masks of the same long context, but physical deletion requires repeated, expensive suffix reconstruction even with prefix caching. We introduce *TraceAttention*, a training-free evaluator that prefills once and reuses the full-context state by masking only target-to-context attention. This preserves the mask-to-score interface of existing attribution estimators while replacing repeated context reconstruction with target-only replay. The intervention differs from deletion, so we evaluate its utility using held-out physical deletions. For reasoning LLMs, scoring the full reasoning and answer exposes the reasoning's context dependence but increases per-mask work. *Dual TraceAttention* (DualTA) addresses this cost by selecting answer-relevant reasoning segments for a compact target. Across 15 model–context settings, TraceAttention and DualTA provide geometric-mean incremental speedups of 9.6× and 14.2× over prefix-cached deletion, respectively, while retaining over 90% of the deletion reference on average for both the linear datamodeling score (LDS) and Top-5 removal utility. On Qwen3.5-27B at a mean context length of 205K tokens, DualTA reduces incremental attribution latency from 841.0 to 13.1 s, a 64.2× speedup.

1 INTRODUCTION

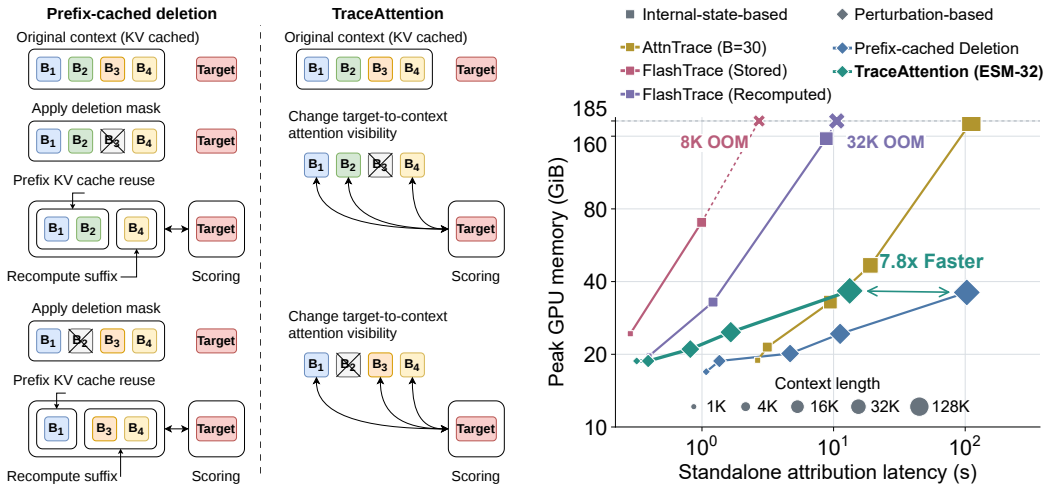
Large language models (LLMs) increasingly operate over long documents (Zhang et al., 2024), interaction histories, and tool outputs (Yang et al., 2024). Context attribution (Cohen-Wang et al., 2024) assigns importance scores to input spans to explain their influence on a given model response. For a document-grounded answer, these spans may be evidence passages; for a tool-using agent, they may be earlier tool results.

Perturbation-based methods (Cohen-Wang et al., 2024; Liu et al., 2025) test this dependence by hiding context blocks and measuring changes in the likelihood of a fixed response. As shown in the left panel of Figure 1(a), these queries share one context and target, creating a repeated long-context inference workload. Even with exact prefix caching (Zheng et al., 2024), physical deletion invalidates states after the first removed block and forces recomputation of the affected suffix. Internal-state-based methods instead derive attribution from layerwise attention or activation interactions, as in AttnLRP (Achtibat et al., 2024), FlashTrace (Pan et al., 2026b), and AttnTrace (Wang et al., 2026). Dense attention maps require quadratic storage if materialized. Modern IO-aware kernels such as FlashAttention (Dao et al., 2022) avoid this materialization, so internal-state-based methods require additional storage, recomputation, or kernel support to access these maps.

We introduce *TraceAttention*, a cache-preserving masked evaluator that exploits this shared query structure. As shown in the right panel of Figure 1(a), TraceAttention prefills the complete context once and reuses its states for any block mask. Each query changes only target-to-context attention visibility and measures target log-likelihood. This replaces repeated context reconstruction with target-only replay, without exporting dense attention maps. The same replay path supports leave-one-out (LOO) (Li et al., 2016), Shapley-style (Lundberg & Lee, 2017), and sampled-mask estimators (Ribeiro et al., 2016; Cohen-Wang et al., 2024). Mechanistic studies of retrieval heads

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.

(Wu et al., 2025) identify attention pathways that mediate long-context retrieval and affect chain-of-thought reasoning. This motivates attention as the intervention site, rather than the attribution score.



(a) Cache-preserving counterfactual evaluation **(b)** Single-GPU latency-memory scaling

Figure 1: TraceAttention execution and single-GPU scaling. (a) Prefix-cached deletion restores the unchanged prefix and recomputes the affected suffix; TraceAttention prefills once and changes target-to-context attention visibility across masks. (b) Standalone attribution latency versus peak GPU memory on one GB200 GPU with Llama-3.1-8B-Instruct. All methods run in PyTorch. Labels identify selected lengths (K = 1,024 tokens); crosses mark the first OOM settings.

Unlike physical deletion, this intervention keeps the full-context state and original token positions fixed. We therefore evaluate the resulting attributions with held-out physical deletions using the linear datamodeling score (LDS) (Park et al., 2023) and Top-k removal utility.

For reasoning LLMs, target choice also changes the attribution signal and per-mask work. FlashTrace (Pan et al., 2026b) shows that attribution for a final answer can concentrate on intermediate reasoning tokens. In TraceAttention, scoring only the answer leaves the realized reasoning in the shared prefix, whereas scoring the complete reasoning-and-answer trajectory recomputes reasoning and answer states under every mask. We introduce *Dual TraceAttention* (DualTA), which identifies answer-relevant reasoning segments and evaluates a compact reasoning-and-answer target to reduce repeated target work.

Across 15 model–context settings, TraceAttention and DualTA provide geometric-mean incremental speedups of 9.6× and 14.2×. They retain 92.7% and 91.5% of deletion-reference LDS on average, and over 94% of Top-5 removal utility. On Qwen3.5-27B at a mean context length of 205K tokens, TraceAttention reduces incremental attribution latency from 841.0 to 28.6 s, and DualTA further reduces it to 13.1 s (64.2× over prefix-cached deletion). As shown in Figure 1(b), TraceAttention's speedup over prefix-cached deletion grows from 3.4× at 1K to 7.8× at 128K with comparable peak memory. At 128K, it is also 8.4× faster than AttnTrace and uses 80.0% less peak memory.

Our contributions are:

- We factor perturbation-based context attribution into an estimator and a masked evaluator, and introduce TraceAttention, a training-free evaluator that restricts each mask to target-to-context attention, preserving the full-context state and eliminating repeated context reconstruction.
- We show that the target boundary jointly controls the attribution signal and target-side work in reasoning LLMs, and introduce DualTA to reduce repeated target work.
- Across models, mask budgets, and mean context lengths up to 205K, we demonstrate substantial speedups while retaining most deletion-grounded utility. Single-GPU speedup grows with context length at peak memory comparable to prefix-cached deletion.

2 CONTEXT ATTRIBUTION AS MASKED MODEL EVALUATION

Perturbation-based context attribution evaluates the model under multiple context masks. The attribution estimator selects masks and combines their returned scores into block attributions; the masked evaluator executes each query. This separation lets us change how queries run without changing the estimator, whether the blocks are tokens, sentences, passages, or application-specific spans.

2.1 MASK-TO-SCORE FORMULATION

Let $C = (c_1, \ldots, c_{\ell_C})$ be a context token sequence with n non-overlapping maskable blocks $C_1, \ldots, C_n$, and let $T = (t_1, \ldots, t_{\ell_T})$ be the fixed target tokens whose context dependence we wish to explain. Here, ℓ_C and ℓ_T count tokens, while n counts blocks.

Each query specifies an unmasked conditioning prefix P, a scored target T, and a mask m over designated blocks in P. The prefix contains all tokens before the scored target. For context attribution, $m \in \{0,1\}^n$ retains C_i when $m_i = 1$ and hides it when $m_i = 0$; tokens outside these blocks remain fixed. Each attribution task fixes the prefix, target, and maskable blocks; only the binary mask varies across queries.

Physical deletion removes the hidden blocks to form P_m, preserving the order of the remaining tokens. We write C_m for this operation on C. The evaluator scores the fixed target by its mean log-likelihood (Monteiro Paes et al., 2025):

$$v_{\mathrm{del}}(m; P, T) = \frac{1}{\ell_T} \sum_{k=1}^{\ell_T} \log p_\theta(t_k \mid P_m, t_{<k}). \tag{1}$$

Here, θ denotes fixed model parameters and $t_{<k}$ the preceding target tokens. Each query scores the same target through teacher forcing rather than generating a new response. Target length affects work per query, not the number of queries.

LOO (Li et al., 2016) hides one block at a time, Shapley-style methods (Lundberg & Lee, 2017) average marginal contributions over block subsets, and sampled-mask methods (Ribeiro et al., 2016; Cohen-Wang et al., 2024) infer block scores from multi-block masks. These estimators use the same mask-to-score interface.

2.2 TARGET DECOMPOSITION IN REASONING LLMS

A reasoning-enabled response consists of a realized reasoning trace T_R followed by a final answer or tool action T_A. The target boundary determines whether we score only the answer or also the reasoning that led to it. Table 1 specifies the prefix and target for each choice, with $\oplus$ denoting token-sequence concatenation.

Table 1: Target choices for context attribution. In every row, masks act only on C. The direct response T_A^{dir} is generated with reasoning disabled; $A \mid R$ and $R \oplus A$ share the same realized reasoning-and-answer response.

Target choice	Prefix P	Scored target T
A_{dir}	C	T_A^{dir}
$A \mid R$	$C \oplus T_R$	T_A
$R \oplus A$	C	$T_R \oplus T_A$

Under $A \mid R$, the reasoning belongs to the conditioning prefix and only answer-token likelihoods enter the score. Under $R \oplus A$, reasoning-token likelihoods also enter the score. Both settings keep the same generated tokens fixed. Physical deletion recomputes reasoning states affected by context removal in either setting, even when only the answer is scored. A_{dir} uses a separately generated response rather than a different target boundary on the same response.

3 TRACEATTENTION

TraceAttention prefills the context once and evaluates each mask by changing only target-to-context attention during target replay. This eliminates repeated context reconstruction while preserving the mask-to-score interface in Equation 1.

3.1 CACHE-PRESERVING ATTENTION-VISIBILITY INTERVENTION

To reuse the prefix across masks, TraceAttention leaves its computation unchanged and intervenes only during target replay. Let H_P denote the reusable state obtained by prefilling P, including key-value (KV) caches and any recurrent prefix states. For each mask, target queries cannot attend to prefix tokens in hidden blocks at any self-attention layer. All other connections follow the model's original attention pattern. The cached prefix state remains fixed, while target states are recomputed under each mask.

We write $p_{\theta,m}$ for the token distribution produced by this masked replay with unchanged model parameters. TraceAttention measures target likelihoods rather than attention weights:

$$v_{\mathrm{TA}}(m; P, T) = \frac{1}{\ell_T} \sum_{k=1}^{\ell_T} \log p_{\theta,m}(t_k \mid H_P, t_{<k}). \tag{2}$$

The prefix and target follow Table 1. For $A \mid R$, the realized reasoning remains in the shared prefix $H_{C \oplus T_R}$; for $R \oplus A$, it is replayed from H_C under each context mask. Unlike physical deletion, TraceAttention keeps the prefix states and token positions fixed across masks, so information already encoded in the shared state may remain available. Appendix E.1 gives the attention-level formulation.

3.2 REASONING-GUIDED DUAL TRACEATTENTION

Scoring $T_R \oplus T_A$ exposes context dependence in the reasoning itself, but replays the entire reasoning trace for every context mask. Dual TraceAttention first identifies answer-relevant reasoning segments, then uses them to construct a shorter context-scoring target.

Stage 1: Select reasoning for replay. We split T_R into segments $G_1, \ldots, G_J$ at deterministic newline and punctuation boundaries. With the context visible, a mask $z \in \{0,1\}^J$ hides all tokens in G_j from answer queries when $z_j = 0$. Each query uses $v_{\mathrm{TA}}(z; C \oplus T_R, T_A)$, and the estimator combines these scores into segment attributions ψ_j.

Stage 1 measures influence on the answer, not the reasoning's own context dependence. We therefore prioritize retained length to preserve more of the original reasoning for context scoring. A 0–1 knapsack selects whole segments with $\psi_j \geq 0$ under a B-token budget, first maximizing token count and then total score among equally filled selections. We concatenate the selected segments in their original order to form $\widetilde{T}_R$. Appendix E.2 gives the selection objective.

Stage 2: Attribute the context with a compact target. We restore the context-only state H_C and score $\widetilde{T}_R \oplus T_A$ under each context mask m, using $v_{\mathrm{TA}}(m; C, \widetilde{T}_R \oplus T_A)$. Selection is performed once, while the shorter target reduces replay work across all context-mask queries.

3.3 COMPUTATIONAL COST

TraceAttention removes repeated context reconstruction, and DualTA reduces the remaining target replay. For dense-attention Transformer blocks of width d and fixed depth and feed-forward expansion ratio, replaying ℓ_T target tokens against ℓ_C cached context tokens requires

$$\mathcal{O}\left(\underbrace{\ell_T d^2}_{\text{projections and MLPs}} + \underbrace{\ell_C \ell_T d}_{\text{target–context attention}} + \underbrace{\ell_T^2 d}_{\text{target–target attention}} \right). \tag{3}$$

Physical deletion reuses exact cached prefixes but must reconstruct the uncached retained suffix. When that suffix contains a number of tokens proportional to ℓ_C, its reconstruction adds $\mathcal{O}(\ell_C d^2 + \ell_C^2 d)$ work per mask. TraceAttention eliminates this repeated context work; its initial prefill is needed only when the prefix state is not already resident.

Let R_C and R_R count context- and reasoning-mask queries, and let $\ell_R = |T_R|$ and $\ell_A = |T_A|$. Full-target TraceAttention replays $\ell_R + \ell_A$ tokens for each context mask. DualTA adds R_R answer-scoring queries in Stage 1, then replays at most $B + \ell_A$ tokens per context mask in Stage 2. Using B to upper-bound retained reasoning, we activate DualTA when the saved reasoning-token replays exceed the answer-token replays added by Stage 1:

$$\underbrace{R_C(\ell_R - B)}_{\text{reasoning-token replays saved}} > \underbrace{R_R \ell_A}_{\text{Stage 1 answer-token replays}} . \tag{4}$$

Otherwise, Stage 1 is skipped and full-target TraceAttention is used. This rule compares replay-token work; all reported speedups use measured wall-clock latency. Appendix E.3 gives the per-method cost expressions.

4 Experiments

We evaluate attribution quality and execution efficiency jointly. Held-out physical deletions define the quality test, and the same estimator fitted to physical-deletion queries at the same mask budget serves as the deletion reference. We report LDS (Park et al., 2023), Top-k removal utility, incremental attribution latency, and peak GPU memory in the single-GPU sweep.

4.1 Experimental Setup

Data and comparisons. The target study uses Qwen3.5-4B (Qwen Team, 2026) on 256 sampled examples per dataset from SWE-agent-trajectories (Nebius, 2024; Yang et al., 2024), AMA-Bench (Zhao et al., 2026), and InfiniteBench (Zhang et al., 2024). Dual TraceAttention is selected on a separate InfiniteBench development subset. Breadth additionally evaluates NVIDIA-Nemotron-3-Nano-4B (NVIDIA Corporation, 2026) and Gemma-4-E4B-it (Gemma Team, 2026) across these sources and HotpotQA (Yang et al., 2018). Scaling uses Qwen3.5-27B and InfiniteBench.

All paired physical-deletion–TraceAttention comparisons differ only in the masked evaluator: physical deletion constructs C_m, whereas TraceAttention reuses the full-context state.

Estimators and metrics. We use example-specific surrogate modeling with R balanced masks (ESM-R) as the primary estimator, fitting a ContextCite-style sparse surrogate (Cohen-Wang et al., 2024). LDS (Park et al., 2023) measures how well summed scores of retained blocks predict target-likelihood rankings across 256 held-out physical deletions. Top-k removal utility measures the decrease in mean target log-likelihood after physically deleting the k highest-scoring blocks, with $k \in \{1, 3, 5\}$. Higher values indicate better attribution quality for both metrics. We report full LDS over all evaluated examples, with undefined correlations contributing zero.

System and timing. All dataset experiments use the same single-stream `llama.cpp` path with BF16 weights, FP16 cache states, CUDA FlashAttention (Dao et al., 2022), GPU-side target NLL, and matched microbatch settings on NVIDIA GB200 GPUs. Physical deletion uses exact prefix caching. We report incremental attribution latency after the required prefix state is resident; average speedups are geometric means of per-setting latency ratios.

We use PyTorch for the single-GB200 systems sweep to compare all methods in the framework used by FlashTrace and AttnTrace. The matched workload uses Llama-3.1-8B-Instruct (Grattafiori et al., 2024) with 512 reasoning tokens, 128 answer tokens, and 128-token context blocks. Physical deletion and TraceAttention use ESM-32; AttnTrace uses 30 context subsamples with a FlashAttention-2 (Dao, 2024) forward pass. FlashTrace retains its official eager-attention setting for attention-weight access.

4.2 Attribution Targets Across Datasets

We study how the attribution target affects TraceAttention using a fixed ESM-32 estimator and Qwen3.5-4B on the three datasets. For each dataset, we compare A_{dir}, $A \mid R$, and $R \oplus A$ on the same sampled source examples (Figure 2(a)).

The $A \mid R$ and $R \oplus A$ conditions share the same generated reasoning and answer and differ only in the target boundary; A_{dir} is generated separately with reasoning disabled.



Figure 2: Attribution targets and Dual TraceAttention with Qwen3.5-4B. (a) Physical-deletion and TraceAttention LDS with ESM-32 across the three target choices and datasets. Each target choice uses its corresponding physical-deletion evaluation target. (b) LDS on $R \oplus A$ versus incremental attribution latency for Deletion-ESM-32, TraceAttention-ESM-32, and Dual TraceAttention across Stage 1 configurations and reasoning budgets; the random-segment control is shown at $B = 256$.

Reasoning-inclusive targets. Across all three datasets in Figure 2(a), $R \oplus A$ yields the highest mean LDS and $A \mid R$ the lowest for both physical deletion and TraceAttention. The separation is largest on InfiniteBench, where TraceAttention obtains 0.806 LDS under $R \oplus A$, compared with 0.519 under A_{dir} and 0.305 under $A \mid R$. The low LDS under $A \mid R$ is consistent with the information-absorption effect (Pan et al., 2026b) reported for reasoning tokens. Scoring the complete trajectory includes the context dependence of reasoning itself, rather than only of the final answer.

Deletion-reference agreement. Reasoning-inclusive scoring also narrows the LDS gap between TraceAttention and physical deletion most clearly on InfiniteBench. The LDS gap in Table 6 falls from 0.294 under $A \mid R$ to 0.016 under $R \oplus A$, with TraceAttention retaining 98.1% of reference LDS and 99.8% of Top-5 removal utility. Larger gaps remain on agent histories, making the gain in agreement task-dependent. Scoring the longer target also increases replay cost: on InfiniteBench (Table 2), the mean scored target grows from 53 to 3,250 tokens and TraceAttention latency from 0.39 to 11.14 s per example, motivating the compact targets studied next.

4.3 Dual TraceAttention: Quality–Cost Trade-off

We evaluate how much full-target attribution signal a compact reasoning target retains. On a Qwen3.5-4B InfiniteBench development subset, Stage 1 attributes reasoning segments to T_A, with the context visible, using TraceAttention-ESM-R_R for $R_R \in \{32, 64, 128\}$ or TraceAttention-LOO; Stage 2 uses TraceAttention-ESM-32. We vary $B \in \{128, 256, 512\}$, evaluate LDS on the complete $R \oplus A$ target and incremental attribution latency, and fix the selected configuration thereafter (Figure 2(b)).

Reasoning-aware compaction. The selected configuration, Stage 1 ESM-64 with $B = 512$ (Figure 2(b)), obtains 0.776 LDS versus 0.806 for full-target TraceAttention (96.3% retention) and 0.714 for random selection at the same reasoning budget (Table 13). The guided policy also reduces undefined correlations from 11.6% to 2.6% (Appendix C.3). On activated examples (Table 14), it retains 512 of 3,795 reasoning tokens (13.5%), substantially reducing the work repeated for each context mask.

Quality–cost frontier. At $B = 512$ in Figure 2(b), ESM-64 improves LDS over ESM-32 (0.776 versus 0.767) with modest added latency (2.58 versus 2.34 s). Doubling the Stage 1 query count to ESM-128 provides no further mean LDS gain, making ESM-64 the selected quality–cost operating point. Including selection overhead, this point is 28.1× faster than physical deletion and 4.3× faster than full-target TraceAttention.

4.4 TraceAttention Across Reasoning Models and Contexts

We evaluate full-target TraceAttention and DualTA across five context settings and three model architectures: local/global attention, Mamba-2 hybrid, and Gated DeltaNet hybrid (Figure 3). The hybrid models also carry context information through recurrent states. We leave this path intact and test whether self-attention masking still yields useful context attribution. DualTA uses the selected Stage 1 ESM-64 with $B = 512$. All quality metrics use physical deletions scored on the complete $R \oplus A$ target as the common evaluation; the deletion reference and both context estimators use ESM-32. Appendix D shows selected reasoning segments and context sources.

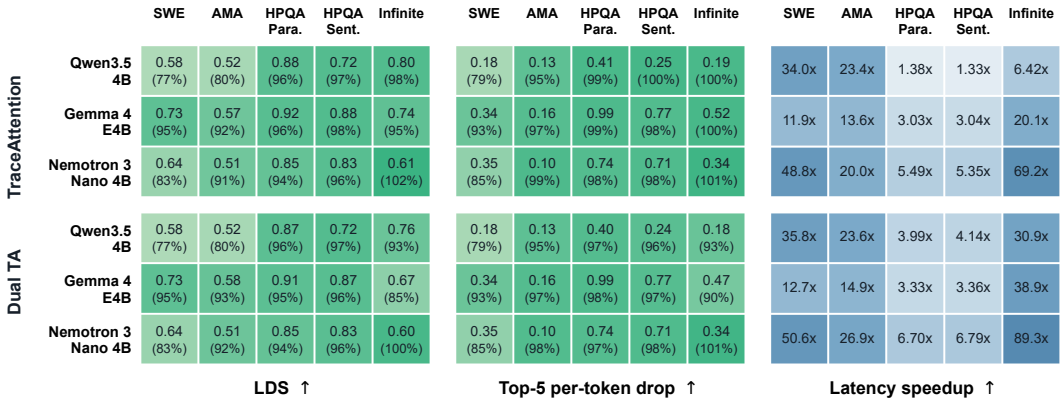


Figure 3: Full-target TraceAttention (top) and DualTA (bottom) across three reasoning models and five context settings. LDS and Top-5 cells show absolute values, with retention relative to the matched complete-$R \oplus A$ deletion reference in parentheses. Latency cells show incremental speedup over prefix-cached deletion. DualTA uses Stage 1 ESM-64 with $B = 512$ throughout.

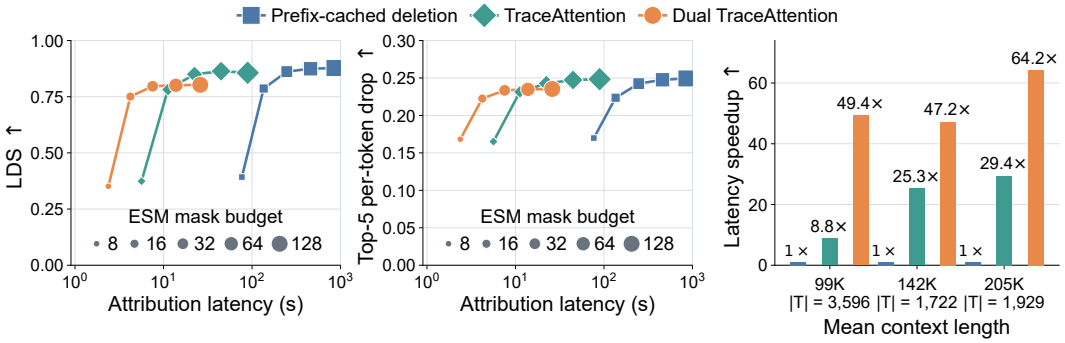
Attribution quality. Across the settings in Figure 3, TraceAttention preserves useful attribution even with recurrent prefix states left intact. TraceAttention retains 92.7% of reference LDS and 96.0% of Top-5 utility on average; DualTA retains 91.5% and 94.3%, respectively (Table 7). LDS retention is highest on HotpotQA and InfiniteBench, with larger gaps on Qwen agent histories and Nemotron SWE-agent-trajectories.

Serving-time efficiency. Across the 15 settings in Figure 3, TraceAttention and DualTA provide geometric-mean incremental speedups of 9.6× and 14.2×, respectively (Table 8). Target compaction is useful even with short contexts: on Qwen sentence-level HotpotQA, it reduces latency from 2.82 to 0.90 s while LDS changes by less than 0.001 (Table 7). On Qwen InfiniteBench, speedup over deletion rises from 6.4× to 30.9× as LDS changes from 0.803 to 0.761. The additional gain therefore tracks target replay as well as context length.

4.5 Mask-Budget Scaling and Long-Context Evaluation

We study mask-budget scaling and long-context attribution using Qwen3.5-27B and InfiniteBench. For mask-budget scaling, we fix the example subset and vary the number of ESM mask queries. For the long-context evaluation, we fix $R_C = 32$ and evaluate three settings with mean context lengths of 99K, 142K, and 205K tokens after tokenization.

We compare prefix-cached physical deletion, full-target TraceAttention, and Dual TraceAttention in Figure 4. Dual uses the reasoning budget and Stage 1 configuration selected in Section 4.3; no parameter is retuned in this section.



(a) LDS **(b) Top-5 drop** **(c) Long-context speedup**

Figure 4: Quality–cost evaluation with Qwen3.5-27B on InfiniteBench. (a–b) LDS and Top-5 log-probability drop versus incremental attribution latency across ESM mask budgets. (c) Incremental speedup over prefix-cached deletion at fixed $R_C = 32$ and the selected DualTA configuration. Labels show mean context and full-target lengths (K = 1,000 tokens).

4.5.1 Mask-Budget Pareto Frontier

On this fixed subset, we vary $R \in \{8, 16, 32, 64, 128\}$; Dual keeps $R_R^\star$ and $B^\star$ fixed and uses ESM-R in Stage 2.

In Figure 4(a–b), increasing the mask budget improves both LDS and Top-5 drop sharply through $R = 32$, after which the gains largely saturate. At every matched budget in Table 9, full-target TraceAttention retains over 95% of reference LDS and over 97% of Top-5 removal utility, with 9.6–13.6× incremental speedups over physical deletion. This enables more mask queries and higher LDS at lower latency: TraceAttention-ESM-32 obtains 0.850 LDS in 22.4 s per example, whereas Deletion-ESM-8 obtains 0.392 LDS in 76.8 s.

Dual TraceAttention obtains 0.796 LDS in 7.55 s per example at $R = 32$ (Table 9), corresponding to a 32.7× speedup over deletion. Further increases yield diminishing returns, so we select $R^\star = 32$ as the knee of the quality–latency curve for the long-context evaluation.

4.5.2 Long-Context Attribution

We fix ESM-$R^\star$ and the selected Dual configuration; Equation 4 selects the path per example.

Across the three context settings in Table 10, TraceAttention retains over 98% of reference LDS and over 99% of Top-5 removal utility, with 8.8–29.4× incremental speedups over physical deletion. Dual TraceAttention retains 87.1–93.4% of reference LDS and over 93% of Top-5 utility, with 47.2–64.2× speedups (Figure 4(c)).

Target length also affects replay cost. From the 99K to the 142K setting in Table 10, the mean full target shrinks from 3,596 to 1,722 tokens, and TraceAttention latency falls from 39.6 to 20.7 s despite the longer context.

Single-GPU scaling. On the matched single-GB200 workload in Figure 1(b), TraceAttention's memory–latency trade-off becomes increasingly favorable with context length. At 4K (Table 11), it is faster than both stored- and recomputed-attention FlashTrace (0.39 versus 0.99 and 1.21 s), with lower peak memory (18.8 versus 70.4 and 32.9 GiB). At 16K, recomputed FlashTrace takes 10.8× as long and uses 7.4× the peak memory of TraceAttention. Stored and recomputed FlashTrace run out of memory at 8K and 32K, respectively. At 128K, TraceAttention is 8.4× faster than AttnTrace and uses 80.0% less peak memory (13.3 versus 111.2 s; 36.6 versus 183.5 GiB). Its speedup over prefix-cached deletion grows from 3.4× at 1K to 7.8× at 128K with comparable peak memory. These standalone timings include TraceAttention's initial context prefill, which is avoided when attribution reuses a resident KV cache.

5 Related Work

Efficient LLM execution and state reuse. FlashAttention (Dao et al., 2022) and PagedAttention (Kwon et al., 2023) optimize attention IO and KV-cache management, respectively, within inference requests. SGLang (Zheng et al., 2024) reuses matching prefixes across calls but computes uncached suffixes. Prompt Cache (Gim et al., 2024) independently encodes reusable prompt modules. In its FiD path, TextGenSHAP (Enouen et al., 2024) relies on independent document encoding to reuse representations across subsets by changing decoder cross-attention. TraceAttention instead preserves cross-block attention during prefill and reuses the full-context state across masks through target-to-context self-attention masking.

Context attribution. Removal-based explanation frameworks (Covert et al., 2021) distinguish the removal mechanism, explained behavior, and attribution summary; MExGen (Monteiro Paes et al., 2025) extends perturbation-based attribution to text generation at multiple input granularities. CAGE (Walker & Ewetz, 2025) and Tokengeist (Tang et al., 2026) trace indirect contributions through generated tokens and conversation turns, respectively. ContextCite (Cohen-Wang et al., 2024) samples binary context masks and fits an example-specific sparse surrogate; AttriBoT (Liu et al., 2025) reduces approximate-LOO cost through prefix reuse, hierarchy, and proxy models; and ARC-JSD (Li et al., 2026) scores context sentences using response-distribution divergence under physical ablation. TracLLM (Wang et al., 2025) uses informed search, while CAMAB (Pan et al., 2026a) adaptively selects context subsets through bandit optimization. Physical deletion still reconstructs uncached context states, even with fewer queries. TraceAttention eliminates this work by reusing one full-context state across single- and multi-block masks. Its mask-to-score interface lets estimators use this execution path without changing their mask-selection or aggregation rules.

Attention perturbation. Thought Anchors (Bogdan et al., 2025) studies reasoning-step importance through resampling and attention masking. AtMan (Deiseroth et al., 2023) measures output changes after attention perturbations. In its decoder-only formulation, these perturbations also affect context-to-context attention, so context states change across queries. TraceAttention instead masks only target-to-context attention, preserving the full-context state and eliminating per-mask context reconstruction.

Internal-state-based attribution. Hidden-state span matching (Phukan et al., 2024) has also been studied for contextual question answering. Other methods construct attribution directly from gradients (Sundararajan et al., 2017), observed attention (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019), or relevance propagation (Achtibat et al., 2024). AT2 (Cohen-Wang et al., 2025) learns an attention-based predictor from ablation labels; FlashTrace (Pan et al., 2026b) aggregates contributions over reasoning spans; and AttnTrace (Wang et al., 2026) combines attention scores with context subsampling. Attention-map access requires additional storage, recomputation, or attribution-specific kernel support. TraceAttention instead scores target likelihoods through target-only replay without exporting dense attention maps.

6 Conclusion and Limitations

We introduced TraceAttention for efficient perturbation-based long-context attribution. It reuses one full-context state across masks, eliminating repeated context reconstruction while preserving the mask-to-score interface. Across 15 model–context settings, TraceAttention and DualTA achieve average retention above 90% for both deletion-reference quality metrics, with geometric-mean incremental speedups of 9.6× and 14.2×, respectively. The single-GB200 sweep further shows a growing latency advantage with context length while peak memory remains close to prefix-cached deletion. We also showed that target choice controls the attribution signal and replay cost, and introduced DualTA to reduce this work using answer-relevant reasoning segments.

TraceAttention keeps full-context states and original token positions fixed while changing target-to-context attention visibility; held-out physical deletions evaluate attribution quality. DualTA additionally assumes an observable reasoning trace and depends on its segmentation and Stage 1 selection.

AI USE STATEMENT

We used generative AI tools throughout the research and writing process, including research ideation, literature discovery, methodological development, and experimental design. Research ideas were developed through iterative discussions in which the authors contributed ideas, guided the research direction, and evaluated and refined AI-generated proposals. AI coding tools generated the method implementation under our direction. Generative AI was also used to draft the full initial manuscript, including its mathematical formulation, and to revise its structure, technical exposition, and wording. The authors determined the research framing and experimental controls, identified the key observations, and led the interpretation of results. AI tools helped articulate and refine these interpretations and identify relevant evidence within the experimental results, with the authors evaluating the suggestions and deciding which to incorporate. The authors reviewed the TraceAttention implementation line by line for consistency with the intended method, prepared and executed the experimental pipelines, and reviewed and revised the AI-assisted manuscript. We take responsibility for the final text, claims, code, and reported results.

ETHICS STATEMENT

This work studies efficient context attribution using publicly available language models and benchmark datasets. We identify no ethical concerns specific to the experiments reported in this paper.

REPRODUCIBILITY STATEMENT

Section 3 specifies TraceAttention and DualTA. Appendix A documents the datasets, preprocessing, target generation, and evaluation settings. Appendix B.5 details the baseline implementations, software environments, and latency and memory measurements. Appendix E provides the attention-level formulation, selection objective, and computational cost analysis. The supplementary material includes our `llama.cpp` and PyTorch implementations, dataset converters, and experiment scripts.

REFERENCES

Reduan Achtibat, Sayed Mohammad Vakilzadeh Hatefi, Maximilian Dreyer, Aakriti Jain, Thomas Wiegand, Sebastian Lapuschkin, and Wojciech Samek. AttnLRP: Attention-aware layer-wise relevance propagation for transformers. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 135–168. PMLR, 2024.

Paul C. Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. Thought Anchors: Which LLM reasoning steps matter? *arXiv preprint arXiv:2506.19143*, 2025. URL https://arxiv.org/abs/2506.19143.

Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. ContextCite: Attributing model generation to context. In *Advances in Neural Information Processing Systems*, volume 37, pp. 95764–95807, 2024.

Benjamin Cohen-Wang, Yung-Sung Chuang, and Aleksander Madry. Learning to attribute with attention. *arXiv preprint arXiv:2504.13752*, 2025.

Ian Covert, Scott Lundberg, and Su-In Lee. Explaining by removing: A unified framework for model explanation. *Journal of Machine Learning Research*, 22(209):1–90, 2021. URL https://jmlr.org/papers/v22/20-1316.html.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2307.08691.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*, volume 35, pp. 16344–16359, 2022.

Björn Deiseroth, Mayukh Deb, Samuel Weinbach, Manuel Brack, Patrick Schramowski, and Kristian Kersting. ATMAN: Understanding transformer predictions through memory efficient attention manipulation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 63437–63460, 2023.

Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. Flex Attention: A programming model for generating optimized attention kernels. *arXiv preprint arXiv:2412.05496*, 2024. URL https://arxiv.org/abs/2412.05496.

James Enouen, Hootan Nakhost, Sayna Ebrahimi, Sercan Arik, Yan Liu, and Tomas Pfister. TextGenSHAP: Scalable post-hoc explanations in text generation with long documents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13984–14011. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.832. URL https://aclanthology.org/2024.findings-acl.832/.

Gemma Team. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026. URL https://arxiv.org/abs/2607.02770.

In Gim, Guojun Chen, Seung-seob Lee, Nikhil Sarda, Anurag Khandelwal, and Lin Zhong. Prompt Cache: Modular attention reuse for low-latency inference. In *Proceedings of Machine Learning and Systems*, volume 6, 2024.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Sarthak Jain and Byron C. Wallace. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3543–3556. Association for Computational Linguistics, 2019.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626. ACM, 2023. doi: 10.1145/3600006.3613165.

Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016.

Ruizhe Li, Chen Chen, Yuchen Hu, Yanjun Gao, Xi Wang, and Emine Yilmaz. Attributing response to context: A Jensen–Shannon divergence driven mechanistic study of context attribution in retrieval-augmented generation. In *The Fourteenth International Conference on Learning Representations*, 2026.

Fengyuan Liu, Nikhil Kandpal, and Colin Raffel. AttriBoT: A bag of tricks for efficiently approximating leave-one-out context attribution. In *The Thirteenth International Conference on Learning Representations*, 2025.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30, pp. 4765–4774, 2017.

Lucas Monteiro Paes, Dennis Wei, Hyo Jin Do, Hendrik Strobelt, Ronny Luss, Amit Dhurandhar, Manish Nagireddy, Karthikeyan Natesan Ramamurthy, Prasanna Sattigeri, Werner Geyer, and Soumya Ghosh. Multi-level explanations for generative language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32291–32317. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.1553.

Nebius. SWE-agent-trajectories. Hugging Face dataset, 2024. URL https://huggingface.co/datasets/nebius/SWE-agent-trajectories.

NVIDIA Corporation. NVIDIA-Nemotron-3-Nano-4B-BF16. Hugging Face model card, 2026. URL https://huggingface.co/nvidia/NVIDIA-Nemotron-3-Nano-4B-BF16.

Deng Pan, Keerthiram Murugesan, Ting Hua, Nuno Moniz, and Nitesh V. Chawla. Context attribution with multi-armed bandit optimization. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 11651–11662. Association for Computational Linguistics, 2026a. doi: 10.18653/v1/2026.findings-acl.565. URL https://aclanthology.org/2026.findings-acl.565/.

Wenbo Pan, Zhichao Liu, Xianlong Wang, Haining Yu, and Xiaohua Jia. Towards long-horizon interpretability: Efficient and faithful multi-token attribution for reasoning LLMs. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026b.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. TRAK: Attributing model behavior at scale. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 27074–27113. PMLR, 2023.

Anirudh Phukan, Shwetha Somasundaram, Apoorv Saxena, Koustava Goswami, and Balaji Vasan Srinivasan. Peering into the mind of language models: An approach for attribution in contextual question answering. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 11481–11495. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.682.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144. ACM, 2016. doi: 10.1145/2939672.2939778.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3319–3328. PMLR, 2017.

Jessica Tang, Shraddha Barke, and Sharad Agarwal. Tokengeist: Multi-turn attribution tracing in agentic conversations. *arXiv preprint arXiv:2607.22610*, 2026. URL https://arxiv.org/abs/2607.22610.

Chase Walker and Rickard Ewetz. Explaining the reasoning of large language models using attribution graphs. *arXiv preprint arXiv:2512.15663*, 2025. URL https://arxiv.org/abs/2512.15663.

Yanting Wang, Wei Zou, Runpeng Geng, and Jinyuan Jia. TracLLM: A generic framework for attributing long context LLMs. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 3845–3864. USENIX Association, 2025. URL https://www.usenix.org/conference/usenixsecurity25/presentation/wang-yanting.

Yanting Wang, Runpeng Geng, Ying Chen, and Jinyuan Jia. AttnTrace: Contextual attribution of prompt injection and knowledge corruption. In *2026 IEEE Symposium on Security and Privacy*, pp. 4320–4338. IEEE, 2026. doi: 10.1109/SP63933.2026.00198.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 11–20. Association for Computational Linguistics, 2019.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. In *The Thirteenth International Conference on Learning Representations*, 2025.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik R. Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380. Association for Computational Linguistics, 2018.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Hao, Xu Han, Zhen Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. ∞Bench: Extending long context evaluation beyond 100K tokens. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15262–15277. Association for Computational Linguistics, 2024.

Yujie Zhao, Boqin Yuan, Junbo Huang, Haocheng Yuan, Zhongming Yu, Haozhou Xu, Lanxiang Hu, Abhilash Shankarampeta, Zimeng Huang, Wentao Ni, Yuandong Tian, and Jishen Zhao. AMA-Bench: Evaluating long-horizon memory for agentic applications. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 62557–62583, 2024.

A Experimental Details

This section specifies attribution units, dataset sampling, target construction, and execution settings. Context length determines the shared prefix state, while target length determines the work repeated for each mask.

A.1 Datasets and Attribution Units

The datasets provide complementary context structures: software-agent histories, memory from agent-environment interactions, long documents, and multi-hop evidence passages. Each is converted into ordered typed blocks before chat-template rendering. Supported types are system, user, context, assistant, tool exchange, and target. System and target blocks remain fixed; dataset-level maskability flags specify additional fixed blocks. The scored target is a model-generated continuation, with the target boundary chosen as specified in the corresponding experiment.

SWE-agent-trajectories. SWE-agent-trajectories (Nebius, 2024; Yang et al., 2024) contains software-agent histories with model reasoning, actions, and observations from the environment. Our conversion preserves user, assistant, and tool-exchange boundaries as attribution units. The target is a generated continuation of the history; the scored span follows the target choice used in each experiment.

AMA-Bench. AMA-Bench (Zhao et al., 2026) studies memory over agent-environment interactions, whose records contain machine-generated observations and evolving task states. We use its preprocessed memory and context blocks in their original order and attribute the generated response conditioned on those records.

InfiniteBench. InfiniteBench (Zhang et al., 2024) combines naturally long inputs with synthetic tasks that exercise retrieval and processing of information across long contexts. We retain the preprocessed context blocks and task prompt, then attribute the model-generated continuation under the selected target boundary.

HotpotQA. HotpotQA (Yang et al., 2018) questions require evidence from multiple Wikipedia passages. We use paragraph- and sentence-level context partitions and attribute the generated response to those units.

A.2 Models and Prefix-State Representations

The three reasoning models provide complementary ways of storing and accessing prefix information. They test whether the same attention-visibility evaluator yields useful context attribution across different prefix-state representations.

Gemma-4-E4B-it. Gemma-4-E4B-it (Gemma Team, 2026) interleaves sliding-window and global self-attention.

NVIDIA-Nemotron-3-Nano-4B. NVIDIA-Nemotron-3-Nano-4B (NVIDIA Corporation, 2026) combines Mamba-2 and MLP layers with four self-attention layers. Its recurrent states provide a route for prefix information alongside token-level attention.

Qwen3.5-4B. Qwen3.5-4B (Qwen Team, 2026) interleaves three Gated DeltaNet blocks with one gated self-attention block. Its linear-attention state supplies a different recurrent memory mechanism from Mamba-2.

Across these models, the evaluator changes block visibility at token-addressable attention layers and replays the target from shared full-context prefix states.

A.3 Dataset Sampling

Variable-length sampling uses seed 42 and a 240K-token cutoff on the sampler's context-length field. The target study samples 256 examples per dataset. The breadth study uses 256 examples per context setting, with sentence- and paragraph-level HotpotQA sampled separately.

A 576-example InfiniteBench pool is partitioned by record order: records 1–256 form the Dual TraceAttention development set, 257–512 form the breadth set, and 513–576 form the mask-budget set. These three subsets use non-overlapping records from that pool. The long-context evaluation separately samples 64 examples at each nominal length in $\{92\text{K}, 154\text{K}, 216\text{K}\}$, using seed 42 and tolerance 0.15. Reported model-token lengths are measured after chat-template rendering.

A.4 Workload Scale

Table 2 reports the target-study workloads. Valid counts refer to examples with a complete scored target, determined by response termination and target parsing rather than agreement with a reference answer.

Table 2: Qwen3.5-4B target-study workloads and incremental attribution latency on one GB200 GPU. Valid counts are out of 256 sampled examples per dataset. Lengths and perturbable-block counts are means over valid examples. Both evaluators use ESM-32.

Target	Valid	$\overline{\lvert C\rvert}$	$\overline{\lvert T\rvert}$	$\overline{n}$	Latency (s)	
					Deletion	TraceAttention
SWE-agent-trajectories						
A_{dir}	256	20,596	141.7	27.8	14.47	0.31
$A \mid R$	253	20,477	80.2	27.4	14.49	0.21
$R \oplus A$	251	20,552	215.0	27.5	14.53	0.43
AMA-Bench						
A_{dir}	251	41,470	352.3	78.2	36.39	0.79
$A \mid R$	253	41,352	282.5	78.4	36.87	0.65
$R \oplus A$	253	41,352	711.5	78.4	36.80	1.56
InfiniteBench						
A_{dir}	256	122,611	41.3	137.3	68.61	0.35
$A \mid R$	234	122,635	53.2	139.5	72.31	0.39
$R \oplus A$	234	122,635	3,249.5	139.5	72.66	11.14

The Dual TraceAttention development set contains 234 valid trajectories from 256 sampled examples. They average 122,635 context tokens, 3,195 reasoning tokens, 53 answer tokens, and 347.7 reasoning segments. In the breadth workloads reported in Table 3, Nemotron yields 46 valid InfiniteBench targets from 256 sampled examples. All excluded responses give the answer within the reasoning span without the required reasoning–answer delimiter.

A.5 Attribution Setting

ESM-R fits one sparse surrogate per example from balanced block masks. Following ContextCite-style modeling (Cohen-Wang et al., 2024), the response is the length-normalized log-odds of the target-sequence probability. The implementation fits negative log-odds to block-removal indicators, preserving the positive importance orientation used by LDS and Top-k selection. Table 4 lists the mask budgets, regularization, and evaluation settings.

Deletion-grounded quality. For each target choice, P and T follow Table 1, and held-out masks act only on the context blocks. For block attributions $\phi = (\phi_1, \ldots, \phi_n)$ and $Q = 256$ held-out

Table 3: Complete-$R \oplus A$ workloads in the breadth study. Valid counts are out of 256 sampled examples per setting. Context, reasoning, answer, and perturbable-block statistics are means over evaluated examples after rendering and tokenization.

Dataset / granularity	Valid	$\overline{\lvert C \rvert}$	$\overline{\lvert T_R \rvert}$	$\overline{\lvert T_A \rvert}$	$\overline{n}$
Gemma-4-E4B-it					
AMA-Bench	245	36,339	822	309	78.8
HotpotQA (paragraph)	255	1,428	393	5	9.9
HotpotQA (sentence)	255	1,459	382	5	51.0
InfiniteBench	206	114,614	867	34	139.5
SWE-agent-trajectories	256	21,036	480	148	27.8
NVIDIA-Nemotron-3-Nano-4B					
AMA-Bench	238	40,338	707	217	77.3
HotpotQA (paragraph)	168	1,416	147	7	9.9
HotpotQA (sentence)	201	1,435	150	7	50.7
InfiniteBench	46	118,981	502	7	298.6
SWE-agent-trajectories	243	20,207	104	48	26.9
Qwen3.5-4B					
AMA-Bench	253	41,352	428	282	78.4
HotpotQA (paragraph)	256	1,455	1,576	6	9.9
HotpotQA (sentence)	256	1,487	1,663	7	51.1
InfiniteBench	230	123,700	3,695	58	176.3
SWE-agent-trajectories	251	20,552	135	80	27.5

masks $m^{(q)} \in \{0,1\}^n$, LDS is

$$\mathrm{LDS}(\phi; P, T) = \rho_{\mathrm{S}} \left((v_{\mathrm{del}}(m^{(q)}; P, T))_{q=1}^{Q}, \left(\sum_{i: m_i^{(q)}=1} \phi_i \right)_{q=1}^{Q} \right), \tag{5}$$

where ρ_{S} denotes Spearman rank correlation. The summed attributions predict the ordering of model scores under unseen physical deletions. For the index set $\mathcal{I}_k(\phi)$ of the k highest-scoring blocks, define

$$\Delta_k(\phi; P, T) = v_{\mathrm{del}}(\mathbf{1}; P, T) - v_{\mathrm{del}}(m^{-\mathcal{I}_k}; P, T), \qquad k \in \{1, 3, 5\}. \tag{6}$$

Here, $\mathbf{1}$ retains all blocks and $m^{-\mathcal{I}_k}$ hides only the selected blocks. Since v_{del} is mean target log-likelihood, Δ_k is a per-token log-probability drop. Larger values indicate more effective removal sets. For DualTA, both metrics use $P = C$ and $T = T_R \oplus T_A$, so compact targets are evaluated against the original complete response.

Table 4: Common attribution and deletion-grounded evaluation settings.

Setting	Value
Default context-mask budget	$R_C = 32$
Mask-budget sweep	$R_C \in \{8, 16, 32, 64, 128\}$
Mask removal fraction	0.5
LASSO regularization	$\alpha = 0.01$
Held-out deletion masks	$Q = 256$
Top-k evaluation	$k \in \{1, 3, 5\}$
Estimator / held-out seeds	42 / 424242
Selected Stage 1 / reasoning budget	ESM-64 / $B = 512$
Random-selection repetitions	10 per example

Aggregation. LDS is macro-averaged over all examples with a valid scored target. A constant attribution prediction cannot rank the held-out deletions. Undefined correlations contribute zero so that these failures remain in the evaluation. The same zero rule applies when the physical-deletion scores are constant. Finite negative correlations are retained. Top-k drop averages per-token log-probability changes over examples with at least k perturbable blocks, independently of correlation availability. Quality retention is the ratio of aggregate method and reference means on the same evaluation set. Random-selection results average ten repetitions per example.

A.6 Target Generation and Timing

Targets are generated before attribution using the model-specific settings in Table 5 and replayed teacher-forced. For examples valid in both reasoning-enabled conditions, $A \mid R$ and $R \oplus A$ share the same generated tokens; A_{dir} is generated separately with reasoning disabled. Model-specific chat templates group consecutive compatible messages before tokenization. For text answers, we split the response at the reasoning-end marker specified by the model's chat template. When tool-action parsing is used, preceding assistant text forms T_R and the terminal action span forms T_A; direct-response scoring uses the complete generated response.

Table 5: Model-specific response-generation settings. Qwen denotes Qwen3.5-4B and Qwen3.5-27B; the direct-response condition uses 4B only. Gemma and Nemotron denote Gemma-4-E4B-it and NVIDIA-Nemotron-3-Nano-4B. A top-k value of zero disables that filter.

Model / condition	Reasoning	Temp.	Top-k	Top-p	Presence
Qwen, A_{dir}	off	0.7	20	0.80	1.5
Qwen, reasoning enabled	on	1.0	20	0.95	1.5
Gemma	on	1.0	64	0.95	0.0
Nemotron, other datasets	on	1.0	0	0.95	0.0
Nemotron, SWE-agent-trajectories	on	0.6	0	0.95	0.0

All dataset runs use a maximum response length of 131,072 tokens, min-$p = 0$, repetition penalty 1.0, and generation seed 42. Reasoning-enabled runs use a 122,880-token generation budget for reasoning. The Dual TraceAttention selection budget $B = 512$ instead limits the reasoning retained for attribution.

Shared execution settings. Dataset experiments run on the GB200 NVL72 platform with one GPU per worker and independent examples distributed across workers. All native comparisons use BF16 weights, FP16 cache states, CUDA FlashAttention, GPU-side target negative log-likelihood (NLL), and the same `llama.cpp` path. Maximum batch and microbatch sizes are 131,072 and 2,048 tokens; TraceAttention batches contain up to eight mask queries. Native generation and batch processing use eight CPU threads. Physical deletion restores the longest unchanged prefix and recomputes the affected suffix; TraceAttention reuses the full-context state and original token positions.

Timing boundaries. Dataset experiments report incremental attribution latency after the required prefix states are resident. Dual TraceAttention includes executed Stage 1 queries, target selection, and Stage 2 scoring, together with full-target TraceAttention on examples that skip Stage 1. Response generation and held-out quality-evaluation queries are excluded. Latency per example is total incremental attribution time divided by the number of evaluated examples. The single-GPU synthetic sweep includes context prefill and attribution execution after model loading, with mask preparation and warm-up excluded.

B Complete Attribution and Scaling Results

These comparisons use the same attribution estimator with physical deletion and TraceAttention, then examine target selection and execution scaling.

B.1 Target Boundary

Each target choice is evaluated using physical deletions scored on that target. The confidence intervals in Table 6 are paired bootstrap intervals for the mean LDS difference, with the same zero-contribution rule applied before aggregation.

Table 6: Qwen3.5-4B target-boundary results with ESM-32. Top-5 is the mean per-token log-probability drop. The final column reports a paired 95% confidence interval for the mean LDS difference, TraceAttention minus physical deletion.

Target	Physical deletion		TraceAttention		95% CI of LDS gap
	LDS	Top-5	LDS	Top-5	
SWE-agent-trajectories					
A_{dir}	0.693	0.3740	0.571	0.3108	$[-0.148, -0.097]$
$A \mid R$	0.567	0.1019	0.392	0.0856	$[-0.218, -0.134]$
$R \oplus A$	0.749	0.2277	0.579	0.1802	$[-0.206, -0.135]$
AMA-Bench					
A_{dir}	0.589	0.1659	0.475	0.1673	$[-0.142, -0.087]$
$A \mid R$	0.512	0.0848	0.386	0.0794	$[-0.167, -0.087]$
$R \oplus A$	0.654	0.1363	0.520	0.1298	$[-0.166, -0.103]$
InfiniteBench					
A_{dir}	0.679	1.4171	0.519	1.3015	$[-0.201, -0.121]$
$A \mid R$	0.599	0.0142	0.305	0.0118	$[-0.349, -0.241]$
$R \oplus A$	0.822	0.1993	0.806	0.1989	$[-0.026, -0.008]$

Reasoning and target replay. Complete-trajectory scoring in Table 6 yields the highest mean LDS for both evaluators across the three datasets. Its cost is most visible on InfiniteBench (Table 2): scoring 3,250 rather than 53 target tokens increases TraceAttention latency from 0.39 to 11.14 s. Deletion changes much less (72.31 to 72.66 s), because its answer-only evaluation already recomputes reasoning states affected by context deletion.

Preserving useful removal sets. In Table 6, TraceAttention preserves 95.2% of deletion-reference Top-5 utility on AMA-Bench with $R \oplus A$, alongside 79.5% LDS retention. Thus, its selected blocks still produce nearly the same likelihood drop when removed, even when prediction across held-out masks is less closely preserved. On SWE-agent-trajectories, retention is 79.1% for Top-5 and 77.3% for LDS, indicating a larger change in both selected-block utility and mask-response prediction. The two metrics distinguish these behaviors rather than treating all ranking differences as equally consequential.

B.2 Transfer Across Models and Contexts

The breadth study fixes Stage 1 ESM-64 with $B = 512$ across all models and context settings. Physical deletion and full-target TraceAttention use ESM-32, as does Stage 2 of Dual TraceAttention. Quality is evaluated on physical deletions scored with the complete $R \oplus A$ target, including for compact-target attribution.

Cross-setting summaries. For each setting, quality retention is the ratio of method and deletion-reference metric means in Table 7. The deletion reference is itself a finite-budget attribution estimate, not a quality upper bound. Retention can therefore exceed 100%. Speedup in Table 8 is the ratio

of total incremental attribution times for physical deletion and the method over the same evaluated examples. We report arithmetic-mean quality retention and geometric-mean speedup across the 15 settings, giving each setting equal weight.

Table 7: Deletion-grounded quality in the breadth study. All context estimators use ESM-32; Dual TraceAttention uses Stage 1 ESM-64 with $B = 512$. Top-5 is the mean per-token log-probability drop. The final row averages per-setting retention ratios and is reported in percent, rather than absolute metric units.

Dataset / granularity	Physical deletion		TraceAttention		Dual TraceAttention	
	LDS	Top-5	LDS	Top-5	LDS	Top-5
Gemma-4-E4B-it						
AMA-Bench	0.617	0.1628	0.567	0.1577	0.577	0.1586
HotpotQA (paragraph)	0.951	1.0049	0.917	0.9915	0.908	0.9888
HotpotQA (sentence)	0.898	0.7906	0.879	0.7725	0.865	0.7667
InfiniteBench	0.780	0.5173	0.739	0.5157	0.665	0.4662
SWE-agent-trajectories	0.765	0.3670	0.730	0.3401	0.728	0.3400
NVIDIA-Nemotron-3-Nano-4B						
AMA-Bench	0.558	0.1037	0.509	0.1024	0.513	0.1015
HotpotQA (paragraph)	0.907	0.7576	0.852	0.7390	0.852	0.7386
HotpotQA (sentence)	0.862	0.7282	0.830	0.7140	0.829	0.7137
InfiniteBench	0.599	0.3348	0.613	0.3398	0.599	0.3384
SWE-agent-trajectories	0.769	0.4163	0.635	0.3522	0.636	0.3522
Qwen3.5-4B						
AMA-Bench	0.654	0.1363	0.520	0.1298	0.520	0.1298
HotpotQA (paragraph)	0.913	0.4156	0.876	0.4133	0.873	0.4039
HotpotQA (sentence)	0.743	0.2501	0.724	0.2507	0.724	0.2399
InfiniteBench	0.821	0.1892	0.803	0.1884	0.761	0.1762
SWE-agent-trajectories	0.749	0.2277	0.579	0.1802	0.579	0.1802
Mean retention (%)	100.0	100.0	92.7	96.0	91.5	94.3

Model-dependent gaps on agent histories. The same context source yields different retention across models in Table 7. On SWE-agent-trajectories, full-target TraceAttention retains 95.4% of reference LDS with Gemma, compared with 82.6% with Nemotron and 77.3% with Qwen. For AMA-Bench, all three models retain at least 95.2% of Top-5 utility, while LDS retention ranges from 79.5% to 91.8%. Thus, the ability to select effective removal sets can be better preserved than overall held-out mask-response prediction.

Target replay in short contexts. The Qwen HotpotQA settings in Table 3 have approximately 1.5K context tokens but 1.6–1.7K reasoning tokens. Dual TraceAttention reduces attribution latency by 2.9–3.1$\times$ relative to full-target TraceAttention (Table 8), with LDS decreases below 0.004 (Table 7). These examples expose target-side cost even when the context is short. In contrast, Gemma and Nemotron produce shorter mean reasoning traces on these settings and activate the two-stage path less frequently (Table 15).

Compaction across reasoning models. The same reasoning budget yields different quality–cost trade-offs on long inputs (Tables 7 and 8). On Gemma InfiniteBench, compaction reduces latency from 5.53 to 2.86 s as LDS changes from 0.739 to 0.665. On Qwen InfiniteBench, it reduces 13.15 to 2.73 s while LDS changes from 0.803 to 0.761. Full-target TraceAttention obtains higher LDS in these settings; Dual TraceAttention offers a lower-cost operating point without retuning its selection budget.

Table 8: Incremental attribution latency and speedup on one GB200 GPU per example. Latencies are seconds per evaluated example, measured after the required prefix states are resident. DualTA includes both executed stages and full-target fallback. The last row is the geometric mean of the 15 per-setting speedups.

Dataset / granularity	Physical deletion	TraceAttention		DualTA	
	Latency	Latency	Speedup	Latency	Speedup
Gemma-4-E4B-it					
AMA-Bench	44.508	3.279	13.6×	2.991	14.9×
HotpotQA (paragraph)	1.870	0.617	3.0×	0.561	3.3×
HotpotQA (sentence)	1.915	0.630	3.0×	0.570	3.4×
InfiniteBench	111.147	5.531	20.1×	2.857	38.9×
SWE-agent-trajectories	18.570	1.566	11.9×	1.457	12.7×
NVIDIA-Nemotron-3-Nano-4B					
AMA-Bench	39.267	1.966	20.0×	1.460	26.9×
HotpotQA (paragraph)	1.414	0.258	5.5×	0.211	6.7×
HotpotQA (sentence)	1.427	0.267	5.3×	0.210	6.8×
InfiniteBench	122.818	1.776	69.2×	1.376	89.3×
SWE-agent-trajectories	15.381	0.315	48.8×	0.304	50.6×
Qwen3.5-4B					
AMA-Bench	37.059	1.582	23.4×	1.571	23.6×
HotpotQA (paragraph)	3.594	2.604	1.4×	0.900	4.0×
HotpotQA (sentence)	3.743	2.815	1.3×	0.904	4.1×
InfiniteBench	84.368	13.146	6.4×	2.731	30.9×
SWE-agent-trajectories	14.553	0.428	34.0×	0.406	35.8×
Geometric mean	–	–	9.6×	–	14.2×

B.3 Mask-Budget Scaling

The mask-budget study in Table 9 uses Qwen3.5-27B on the fixed 64-example InfiniteBench subset and varies $R_C \in \{8, 16, 32, 64, 128\}$. All three evaluators use the same examples at each budget; Dual TraceAttention keeps Stage 1 ESM-64 and $B = 512$ fixed while varying Stage 2 queries.

Table 9: Mask-budget results with Qwen3.5-27B on InfiniteBench. All settings use the same 61 valid trajectories from 64 sampled examples. Quality is evaluated on the original complete $R \oplus A$ target. Top-5 is the mean per-token log-probability drop. Time is incremental attribution latency in seconds per example. DualTA uses Stage 1 ESM-64 with $B = 512$.

R_C	Physical deletion			TraceAttention			DualTA		
	LDS	Top-5	Time	LDS	Top-5	Time	LDS	Top-5	Time
8	0.3925	0.1698	76.78	0.3740	0.1651	5.64	0.3514	0.1682	2.38
16	0.7863	0.2236	135.32	0.7818	0.2315	11.22	0.7505	0.2225	4.22
32	0.8617	0.2424	246.93	0.8496	0.2427	22.37	0.7964	0.2335	7.55
64	0.8743	0.2476	458.26	0.8634	0.2473	44.68	0.8007	0.2348	13.81
128	0.8776	0.2494	853.10	0.8563	0.2483	89.28	0.8028	0.2353	26.03

B.4 Long-Context Attribution

The long-context evaluation in Table 10 uses Qwen3.5-27B with Stage 2 ESM-32 and Stage 1 ESM-64 at $B = 512$.

Table 10: Long-context results with Qwen3.5-27B on InfiniteBench. Each setting has 64 valid examples. Lengths are means for the original context and complete target after chat-template rendering. All context estimators use ESM-32; DualTA uses Stage 1 ESM-64 with $B = 512$. LDS and Top-5 use the original complete $R \oplus A$ target. Top-5 is the mean per-token log-probability drop. Latency is incremental attribution time in seconds per example.

$\overline{\lvert C \rvert}$	$\overline{\lvert T \rvert}$	Method	LDS	Top-5	Latency (s)
98,539	3,596	Physical deletion	0.8572	0.3401	348.96
		TraceAttention	0.8416	0.3427	39.62
		DualTA	0.7464	0.3178	7.07
142,326	1,722	Physical deletion	0.7916	0.3864	522.60
		TraceAttention	0.7835	0.3855	20.66
		DualTA	0.7395	0.3627	11.06
204,902	1,929	Physical deletion	0.7993	0.3334	841.04
		TraceAttention	0.8034	0.3359	28.61
		DualTA	0.7210	0.3104	13.09

B.5 Single-GPU Execution Scaling

The synthetic workload uses Llama-3.1-8B-Instruct with 512 reasoning tokens, 128 answer tokens, and 128-token context blocks. Contexts use block-distinct fixed-width encodings and span 1K to 128K, with K denoting 1,024 tokens. The 128K run reserves space for the response and uses 129,919 context tokens in 1,015 blocks.

Execution paths. The systems sweep uses PyTorch to compare all methods in the framework of the released baselines. All methods load BF16 weights from the same GGUF source and use the prepared token sequences. Physical deletion and TraceAttention use BF16 KV states and the same ESM-32 masks with seed 42. Both use FlashAttention (Dao et al., 2022) for prefix computation and FlexAttention (Dong et al., 2024) for target scoring. Deletion reuses exact shared prefixes through a prefix trie. TraceAttention evaluates eight masks per microbatch with shared context KV pages and separate target KV pages for each mask. Prefill chunks contain at most 32,768 tokens, and target-logit chunks contain at most 128 tokens across the microbatch.

Attribution implementations. FlashTrace 0.1.1 uses its released `ifr-span` routine with 128-token source chunks and 32-token sink chunks, in stored- and recomputed-attention modes. We retain its official eager-attention setting because its attribution routine requires attention weights. Its target convention appends one EOG token to the prepared response. AttnTrace uses FlashAttention-2 for the forward pass and the unmodified per-layer attention reconstruction routine at revision `d006ac7`. It evaluates 30 context subsamples with $q = 0.4$, averages attention across layers, heads, and target positions, and scores each block using its five highest-scoring tokens, or all tokens when fewer than five are present. TraceAttention requires only target likelihoods and applies visibility masks within fused attention, without storing or reconstructing attention maps.

Software and measurements. The experiments use PyTorch 2.7.1 with CUDA 12.8 wheels. Physical deletion, TraceAttention, and FlashTrace use Transformers 5.8.1; AttnTrace uses Transformers 4.46.3 and `flash-attn` 2.8.3.post1. Models run in evaluation mode. Each completed setting receives two full warm-up runs followed by five timed executions. We report median latency and median per-run peak GPU memory in Table 11. Each timed execution includes context prefill and attribution, including surrogate fitting for physical deletion and TraceAttention. Model loading, mask preparation, and warm-up are excluded. CUDA is synchronized around timed execution. Peak GPU memory is sampled from device-wide used memory, including weights, caches, and temporary allocations. After a method runs out of memory, its larger settings are not run.

Native execution. The dataset experiments use `llama.cpp` for a common execution path across the evaluated architectures. Table 12 reports both evaluators on the same synthetic workload in this

Table 11: Standalone attribution latency and peak GPU memory on one GB200 GPU with Llama-3.1-8B-Instruct. All methods run in PyTorch. Values are medians over five executions after two full warm-up runs. OOM marks a failed setting; – marks a larger setting not run after that method's first OOM. K denotes 1,024 tokens.

Context	Physical deletion	TraceAttention	FlashTrace stored	FlashTrace recomputed	AttnTrace
(a) Standalone latency (s)					
1K	1.077	0.319	0.285	0.396	2.650
2K	1.104	0.334	0.509	0.569	2.496
4K	1.356	0.390	0.994	1.213	3.134
8K	2.309	0.511	OOM	2.973	5.188
16K	4.675	0.815	–	8.799	9.456
32K	11.216	1.655	–	OOM	19.038
64K	32.223	4.313	–	–	43.255
128K	102.811	13.256	–	–	111.249
(b) Peak GPU memory (GiB)					
1K	16.93	18.76	24.33	19.68	18.89
2K	18.76	18.76	34.82	22.74	19.68
4K	18.76	18.76	70.37	32.91	21.43
8K	18.77	19.17	OOM	54.43	25.38
16K	20.14	21.01	–	156.28	32.89
32K	24.31	24.69	–	OOM	46.68
64K	28.12	28.72	–	–	78.92
128K	36.07	36.62	–	–	183.52

backend. Both use BF16 weights, FP16 caches, CUDA FlashAttention, and ESM-32; TraceAttention batches up to eight masks. Each entry is one recorded execution, including context preparation after model loading. TraceAttention's speedup grows from 1.9× at 1K to 8.1× at 128K, with 3.1–4.0 GiB additional peak memory across the sweep.

Table 12: Native `llama.cpp` results on the same single-GB200 workload. Each entry is one recorded execution. Latency includes context preparation after model loading.

Context	Latency (s)		Peak memory (GiB)	
	Deletion	TraceAttention	Deletion	TraceAttention
1K	1.527	0.811	15.51	19.47
2K	2.064	0.890	15.86	19.60
4K	3.357	1.043	16.56	19.79
8K	6.498	1.356	17.20	20.33
16K	13.008	2.131	18.28	21.41
32K	29.664	3.989	20.43	23.56
64K	73.556	9.217	24.74	27.88
128K	205.562	25.333	33.20	36.37

C Dual TraceAttention Analysis

The development sweep uses Qwen3.5-4B and the same 234 valid trajectories for every configuration. Stage 2 uses ESM-32. All quality metrics use physical deletion scored on the original complete $R \oplus A$ target, including when Dual TraceAttention fits attributions using a compact target.

C.1 Operating-Point Selection

Table 13 shows how the selection method and reasoning budget affect attribution quality and total latency. Increasing B improves mean LDS within every Stage 1 configuration; at $B = 512$, ESM-64 achieves the highest mean LDS among the compact-target configurations.

Table 13: Complete Dual TraceAttention development sweep. Stage 1 ESM-r uses r reasoning masks; every compact-target row uses Stage 2 ESM-32. The first two rows are full-target references. Latency includes both executed stages and full-target fallback. Random averages ten selections per example. Bold identifies the selected configuration, $R_R = 64, B = 512$.

Stage 1	B	LDS	Top-5	Latency (s)	Activated
Deletion ESM-32		0.8217	0.1993	72.673	–
Full-target TraceAttention ESM-32		0.8058	0.1989	11.136	–
Random	128	0.6153	0.1620	0.775	100.0%
Random	256	0.6621	0.1777	1.166	99.1%
Random	512	0.7140	0.1911	1.996	85.9%
LOO	128	0.6427	0.1646	3.206	95.7%
LOO	256	0.6784	0.1747	3.506	94.4%
LOO	512	0.7102	0.1849	4.102	78.2%
ESM-32	128	0.7151	0.1771	1.165	100.0%
ESM-32	256	0.7269	0.1802	1.551	98.7%
ESM-32	512	0.7666	0.1899	2.343	83.8%
ESM-64	128	0.7039	0.1750	1.437	99.6%
ESM-64	256	0.7330	0.1847	1.819	98.3%
ESM-64	**512**	0.7763	0.1901	2.583	82.1%
ESM-128	128	0.6667	0.1656	1.917	98.7%
ESM-128	256	0.7166	0.1819	2.285	95.7%
ESM-128	512	0.7639	0.1886	3.001	80.3%

Quality retained at the selected point. In Table 13, ESM-64 with $B = 512$ retains 96.3% of full-target TraceAttention's LDS and 95.5% of its Top-5 utility at a 4.3× incremental speedup. Relative to deletion, the corresponding values are 94.5%, 95.4%, and 28.1×. The selected point reduces repeated target work while keeping both quality measures close to the full-target reference.

Budget allocation. At $B = 512$ in Table 13, increasing Stage 1 from ESM-32 to ESM-64 gains 0.0097 LDS for 0.240 additional seconds per example. ESM-128 takes longer without improving either mean quality metric. Keeping ESM-64 but reducing B to 256 saves 29.6% of latency and lowers LDS by 0.0433. This separates the modest benefit of more selection queries from the larger benefit of retaining more reasoning at the selected point.

C.2 Activation and Compaction

Reasoning is split at deterministic newline and punctuation boundaries, including English and CJK delimiters. The selector retains whole segments with non-negative scores, first maximizing token count under B and then total score among equally filled solutions. Selected segments remain in their original order.

The random control assigns positive random segment scores and applies the same selector, using ten seeds derived from base seed 42. It performs no Stage 1 model queries. LOO evaluates one reasoning mask per segment; ESM uses its specified query budget.

Table 14: Reasoning compaction at $B = 512$. Token and segment counts are means within each configuration's activated examples; random selections are additionally averaged over ten repetitions. The activated sets can differ because Stage 1 query costs differ.

Stage 1	Activated	Reasoning tokens		Segments	
		Full	Selected	Full	Selected
Random	201/234	3,655.7	512.0	396.4	175.9
LOO	183/234	3,613.3	430.7	393.4	107.5
ESM-32	196/234	3,735.7	511.8	404.7	56.5
ESM-64	192/234	3,794.6	511.9	411.2	55.7
ESM-128	188/234	3,844.7	511.3	416.9	55.4

Where compaction applies. The selected configuration in Table 14 activates on 192/234 examples, retaining 511.9 of 3,794.6 reasoning tokens (13.5%) in 55.7 of 411.2 segments on average. The remaining 42 examples use full-target TraceAttention without Stage 1. Random selection fills the same token budget with more segments, while LOO leaves more capacity unused.

With $R_C = 32$ and $R_R = 64$, Equation 4 becomes $\ell_R > B + 2\ell_A$. At $B = 256$, 230 examples activate, compared with 192 at $B = 512$ (Table 13). Thus, varying B changes both retained length and path selection; reported quality and latency include the complete executed policy.

C.3 Selection Across Examples

Quality under a fixed replay budget. At $B = 512$ in Table 13, the guided policy raises all-example LDS from 0.7140 for random selection to 0.7763, with similar Top-5 utility (0.1901 versus 0.1911). The guided policy reduces the fraction of undefined correlations from 11.6% to 2.6%. Full LDS includes these cases with zero contribution, so the comparison also accounts for failures to recover a ranking. Including its additional selection work, the guided policy takes 0.59 s more per example than random selection and remains 4.3× faster than full-target TraceAttention.

Allocating work between selection and replay. In Table 13, ESM-32 at $B = 128$ obtains 0.7151 LDS in 1.165 s, compared with 0.6153 for random selection at the same budget. It reaches similar LDS to random selection at $B = 512$ (0.7140 in 1.996 s) with one quarter of the reasoning budget. The larger random target preserves more Top-5 utility (0.1911 versus 0.1771), exposing the separate quality–cost trade-offs of selecting reasoning and replaying it.

C.4 Compaction Across Models and Contexts

The fixed configuration adapts to the realized reasoning length through the activation rule. In Table 15, it selects the two-stage path for 87.5–89.8% of Qwen HotpotQA examples and 88.3% of Qwen InfiniteBench examples. The shorter reasoning traces in Qwen and Nemotron SWE-agent-trajectories activate it in approximately 1% of examples. Token and segment statistics in this table separate the compacted examples from full-target fallback.

Retained reasoning on long inputs. For activated Qwen InfiniteBench examples in Table 15, the compact target retains 511.8 of 4,115.1 reasoning tokens (12.4%) in 54.6 of 418.6 segments on average. The corresponding Gemma setting retains 441.5 of 1,770.2 tokens (24.9%). Because the selector keeps whole non-negative-scored segments, the realized retained length can be below the nominal 512-token capacity.

Table 15: Dual TraceAttention activation and reasoning compaction in the breadth study with Stage 1 ESM-64 and $B = 512$. Activated counts are relative to evaluated examples. Token and segment counts are means over activated examples only; the other examples use full-target TraceAttention without Stage 1.

Dataset / granularity	Activated	Reasoning tokens		Segments	
		Full	Selected	Full	Selected
Gemma-4-E4B-it					
AMA-Bench	55/245	1,500.7	512.0	110.1	32.6
HotpotQA (paragraph)	40/255	746.8	465.1	66.4	39.1
HotpotQA (sentence)	40/255	722.0	453.4	64.0	39.5
InfiniteBench	76/206	1,770.2	441.5	152.9	46.8
SWE-agent-trajectories	29/256	1,384.9	512.0	124.7	43.1
NVIDIA-Nemotron-3-Nano-4B					
AMA-Bench	63/238	1,644.0	508.2	116.3	33.6
HotpotQA (paragraph)	5/168	1,479.0	493.8	118.0	34.8
HotpotQA (sentence)	6/201	1,621.8	493.7	104.8	29.8
InfiniteBench	15/46	973.4	508.3	85.7	40.1
SWE-agent-trajectories	2/243	1,193.0	512.0	75.5	30.5
Qwen3.5-4B					
AMA-Bench	9/253	765.7	508.0	53.3	35.6
HotpotQA (paragraph)	230/256	1,704.5	509.4	178.6	45.5
HotpotQA (sentence)	224/256	1,835.3	509.7	193.5	45.5
InfiniteBench	203/230	4,115.1	511.8	418.6	54.6
SWE-agent-trajectories	3/251	1,615.7	512.0	156.7	40.0

D CASE STUDIES

These examples show which context sources TraceAttention selects and which reasoning segments DualTA retains for context scoring. All use Qwen3.5-4B and ESM-32 context estimators. PE denotes physical deletion, and TA denotes full-target TraceAttention. DualTA uses Stage 1 ESM-64 with $B = 512$ and executes both stages in example B. Within each example, LDS and Top-5 drop use physical deletions scored on the same original complete $R \oplus A$ response.

In the figures, B identifies a context block and G a reasoning segment. Colors show attribution scores normalized by the maximum absolute score within each method and stage. Teal denotes positive scores and coral negative scores; zero remains white. A filled circle marks a Top-5 context block or a retained reasoning segment. Excerpts preserve the original wording, with [...] marking omitted text. Table 16 reports the corresponding results.

Case A: Shared two-hop evidence. PE and TA select both supporting sources in Figure 5: B26 identifies the acquired business, and B4 gives its headquarters. Their other Top-5 selections differ, but TA's removal set produces a larger likelihood drop despite its lower LDS (Table 16).

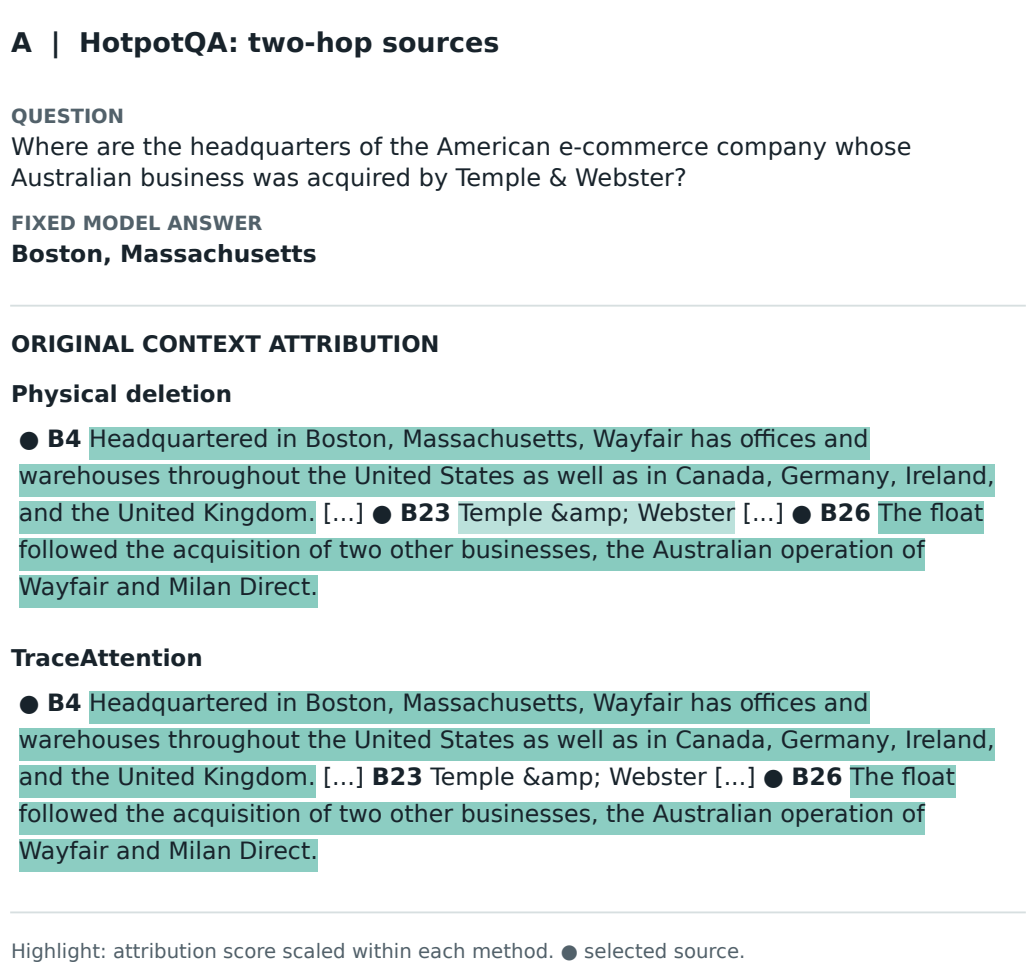


Figure 5: Case A: Context attribution for the Wayfair question from sentence-level HotpotQA (`5ae454d15542996836b02c78`).

Case B: Reasoning selection reduces replay. In Figure 6, DualTA retains the role link in G28 and the full-name evidence in G22 and G151 for context scoring. Using this compact target, DualTA

selects B14 and B9, matching PE and full-target TA. The context-scoring target shrinks from 6,023 to 517 tokens, with higher LDS and Top-5 utility in this example (Table 16).

B | HotpotQA: DualTA reasoning and sources

QUESTION

What was the middle name of David Hayman's mobster role?

FIXED MODEL ANSWER

Gabriel

RETAINED REASONING (EXCERPTS)

● **G22** * Passage "Al Capone": Describes Alphonse Gabriel Capone as an American mobster, crime boss. [...] ● **G28** Mentions he played "Al Capone" at the Citizens Theatre. [...] ● **G151** * If I answer "Gabriel", it comes from the "Al Capone" passage listing his full name.

ORIGINAL CONTEXT ATTRIBUTION

Physical deletion

B5 The film stars David Hayman [...] ● **B9** Alphonse Gabriel Capone [...] was an American mobster, crime boss [...] ● **B14** Hayman played roles such as [...] and Al Capone.

TraceAttention

B5 The film stars David Hayman [...] ● **B9** Alphonse Gabriel Capone [...] was an American mobster, crime boss [...] ● **B14** Hayman played roles such as [...] and Al Capone.

DualTA

● **B5** The film stars David Hayman [...] ● **B9** Alphonse Gabriel Capone [...] was an American mobster, crime boss [...] ● **B14** Hayman played roles such as [...] and Al Capone.

Highlight: attribution score scaled within each method. ● selected source or retained reasoning.

Figure 6: Case B: Retained reasoning and context attribution for the David Hayman question from sentence-level HotpotQA (`5a8e948d5542995085b3745e`).

Case C: Shared sources with lower removal utility. Figure 7 shows that PE and TA both select B12, which identifies `visit_Call`, and B17, which summarizes the reproduction result. PE also selects the test execution in B16, whereas TA selects the failed edit in B13. TA retains 70.1% of PE's LDS, but its complete Top-5 removal set produces a smaller likelihood drop (Table 16).

C | SWE: differing source selections

TASK (EXCERPT)

function call rewrite broken when function is parenthesized [...] (
six
).text_type(x) [...] str(x)

FIXED MODEL CONTINUATION (EXCERPT)

[...] <function=create>
<parameter=filename>
test_fix.py [...]

ORIGINAL CONTEXT ATTRIBUTION

Physical deletion

● **B12** The `visit_Call` method on line 366 is part of the `Py2CompatibleVisitor` class [...] **B13** It appears that my edit attempt failed due to a syntax error, [...] Your changes have NOT been applied. [...] ● **B16** The code has been updated successfully [...] example_text [...] ● **B17** The output of `reproduce_bug.py` is now correctly printing `example_text` [...]

TraceAttention

● **B12** The `visit_Call` method on line 366 is part of the `Py2CompatibleVisitor` class [...] ● **B13** It appears that my edit attempt failed due to a syntax error, [...] Your changes have NOT been applied. [...] **B16** The code has been updated successfully [...] example_text [...] ● **B17** The output of `reproduce_bug.py` is now correctly printing `example_text` [...]

Highlight: attribution score scaled within each method. ● selected source.

Figure 7: Case C: Context attribution before creating `test_fix.py` in SWE-agent-trajectories example `asottile__pyupgrade-166`.

Table 16: Results for cases A–C. Scored tokens count the context-scoring target. Top-5 is the per-token log-probability drop for the complete five-block removal set. Query times cover recorded fitting-mask execution.

Case	Method	Scored tokens	LDS	Top-5	Query time (s)
A	PE	803	0.7695	0.2864	1.997
	TA	803	0.7248	0.3870	1.279
B	PE	6,023	0.6987	0.0868	10.518
	TA	6,023	0.6987	0.0868	10.544
	DualTA	517	0.7620	0.0990	0.802
C	PE	226	0.8727	0.2256	11.333
	TA	226	0.6114	0.1187	0.394

E METHOD AND COST DETAILS

This section gives the attention-level intervention, reasoning selection objective, and computational cost behind Section 3.

E.1 ATTENTION-VISIBILITY INTERVENTION

For a query $v_{\mathrm{TA}}(m; P, T)$, the prefix blocks controlled by m are fixed. At one self-attention head of dimension d_h, let Q_m contain the target-scoring queries and K_m, V_m the keys and values available to those queries. Cached prefix entries are shared across masks; entries produced during target replay are recomputed. The additional visibility mask excludes keys in hidden prefix blocks:

$$[D(m)]_{q,j} = \begin{cases} -\infty, & \text{key } j \text{ belongs to a prefix block hidden by } m, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Let D_{model} be the model's original visibility mask, including causal and any sliding-window restrictions. The attention output is

$$O_m = \mathrm{softmax}\left(\frac{Q_m K_m^\top}{\sqrt{d_h}} + D_{\mathrm{model}} + D(m)\right) V_m. \tag{8}$$

This intervention is applied at every self-attention layer. It changes target access to prefix keys without changing the unmasked prefix computation or token positions across mask queries. In hybrid models, recurrent prefix states are also reused, while recurrent target states are recomputed during replay.

For context attribution, m masks blocks in C, including when $P = C \oplus T_R$ under $A \mid R$. For DualTA Stage 1, $P = C \oplus T_R$ and the mask z acts on reasoning segments rather than context blocks. Stage 2 restores H_C and replays the selected reasoning and answer under context masks.

E.2 REASONING-SEGMENT SELECTION

Let $G_1, \ldots, G_J$ be the reasoning segments and ψ_j their Stage 1 attribution scores. Under token budget B, the selector retains whole segments with non-negative scores:

$$\mathcal{I}_B \in \underset{\substack{\mathcal{I} \subseteq \{j \in \{1,\ldots,J\} : \psi_j \geq 0\} \\ \sum_{j \in \mathcal{I}} |G_j| \leq B}}{\arg\max_{\mathrm{lex}}} \left(\sum_{j \in \mathcal{I}} |G_j|, \sum_{j \in \mathcal{I}} \psi_j \right). \tag{9}$$

The first objective maximizes retained token count; the second maximizes total score among equally filled selections. The compact reasoning preserves the original segment order:

$$\widetilde{T}_R = \bigoplus_{j \in \mathcal{I}_B}^{\text{original order}} G_j. \tag{10}$$

This changes only the target used to fit context attributions. Deletion-grounded evaluation still scores the original complete $T_R \oplus T_A$ response.

E.3 COMPUTATIONAL COST

The following expressions count arithmetic in dense-attention Transformer blocks of width d, with fixed depth and feed-forward expansion ratio. They describe model work rather than wall-clock latency. The output head additionally requires $\mathcal{O}(d|\mathcal{V}|)$ per scored token, where $\mathcal{V}$ is the vocabulary.

Prefix-cached physical deletion. For deletion query r, let $\ell_{P,r} = |P_{m^{(r)}}|$ be the retained prefix length and u_r the number of retained prefix tokens not covered by an available exact cache match. Reconstructing these tokens requires $\mathcal{O}(u_r d^2 + \ell_{P,r} u_r d)$ work. Including target replay, R_C deletion queries require

$$\mathcal{W}_{\mathrm{del}} = \mathcal{O}\left(\sum_{r=1}^{R_C} \left[(u_r + \ell_T) d^2 + \left(\ell_{P,r} u_r + \ell_{P,r} \ell_T + \ell_T^2 \right) d \right] \right). \tag{11}$$

The value of u_r accounts for reuse of any available matching prefix, including prefixes shared across deletion queries. For $A \mid R$, the retained prefix includes T_R, so u_r also includes reasoning states that must be recomputed after deletion.

TraceAttention and DualTA. Let $\ell_R = |T_R|$, $\ell_A = |T_A|$, and $\widetilde{\ell}_R = |\widetilde{T}_R| \leq B$. With the required prefix states resident, full-target TraceAttention replays the reasoning and answer for each context mask:

$$\mathcal{W}_{\mathrm{TA}} = \mathcal{O}\big(R_C(\ell_R + \ell_A)\left[d^2 + (\ell_C + \ell_R + \ell_A)d\right]\big). \tag{12}$$

DualTA instead performs answer replay for reasoning selection, followed by compact-target replay for context attribution:

$$\begin{aligned}\mathcal{W}_{\mathrm{Dual}} = \mathcal{O}\Big(&R_R\ell_A\left[d^2 + (\ell_C + \ell_R + \ell_A)d\right] \\ &+ R_C(\widetilde{\ell}_R + \ell_A)\left[d^2 + (\ell_C + \widetilde{\ell}_R + \ell_A)d\right]\Big).\end{aligned} \tag{13}$$

The Stage 1 term includes attention to the longer prefix $C \oplus T_R$; the Stage 2 term uses the context-only prefix C. Replacing $\widetilde{\ell}_R$ by its upper bound B and comparing replay-token counts gives $R_C(\ell_R + \ell_A) > R_R\ell_A + R_C(B + \ell_A)$, which is equivalent to Equation 4.

Without resident states, TraceAttention also requires one prefill of C. DualTA requires the unmasked states for C and $C \oplus T_R$; the latter can be constructed by extending the former with the realized reasoning. This prefix construction is shared across mask queries rather than repeated for each mask. The expressions above count model evaluation; measured DualTA latency also includes target selection.

Sliding-window and recurrent layers have different arithmetic costs. TraceAttention still avoids rebuilding prefix states for each mask, and DualTA reduces the number of target tokens replayed.