TEXT-ENHANCED MASK PROBABILITY OPTIMIZATION FOR LONG-HORIZON SPAN-TO-SPAN ATTRIBUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Auditing language models that answer from text far earlier in the sequence requires tracing a generated span to the earlier text it depends on, which may lie in a long prompt or in the response itself, far from the span being explained. We formulate this as *long-horizon span-to-span attribution*: the effect of intervening on selected source spans on a fixed later target span, with all other recorded text held unchanged. Propagation-based methods score sources from internal computation and need differentiable or architecture-specific access, while perturbation-based methods measure the target directly but face a combinatorial search whose cost grows with the source. We propose TEMPO (Text-Enhanced Mask Probability Optimization), a black-box method that optimizes a probabilistic mask over source units against the target's likelihood under sampled interventions and decodes the result into a complete ranking across deletion budgets. The decoder uses word and clause structure together with the learned mask probabilities, preserves the best searched set at the search budget, and issues no further model queries, so the number of forward passes is fixed regardless of input length or source–target distance. On four task families spanning retrieval, multi-hop reasoning, mathematical continuation with injected errors and counterfactual multiple choice, TEMPO attains the highest deletion AUC on every task and leads all macro-average faithfulness metrics against seven baselines, at lower cost than the strongest of them. The experiments also show that faithfulness and recovery of annotated evidence diverge: the recall leader changes by task, and a query-matching variant of our pipeline raises reference recall while lowering deletion AUC, indicating that behavioral dependence and annotation agreement must be evaluated separately.

1 INTRODUCTION

A language model's output can depend on text far earlier in the sequence, whether supplied in the prompt or generated earlier in the response. Auditing this process requires tracing a generated claim to the passages it depends on, or an unsafe agent action to the tool-returned text that influenced it (Qi et al., 2024; Wang et al., 2025). Such dependence calls for a *counterfactual* check: perturb a candidate source while holding the output under examination fixed, and measure how its likelihood changes. This is the basis of perturbation-based *faithfulness* evaluation, which tests whether an explanation identifies text that influences model behavior (DeYoung et al., 2020). *Input attribution* addresses this need by identifying which parts of a model's input contribute to a specified output (Cohen-Wang et al., 2024). In autoregressive sequence generation, the question becomes more local: a particular output span is conditioned on both the original input and the text generated before it. We therefore focus on *span-to-span attribution*, relating selected earlier source spans to a later target span by measuring the effect of source perturbations on the fixed target, with the remaining recorded text held unchanged. The source may lie in the prompt or an earlier part of the response, and long inputs or outputs can separate it from the target by many tokens, giving rise to *long-horizon span-to-span attribution*.

Existing attribution methods fall into two families. *Propagation-based* methods derive importance from the model's internal computation, through forward information flow or decomposition (Ferrando & Voita, 2024; Hong et al., 2025) or backward gradient propagation (Sundararajan et al., 2017); a few forward/backward passes score all source units at once, but forward-flow rules are tied to specific architectural components, gradient methods require differentiable access, and internal contribution or sensitivity scores need not predict the effect of finite interventions on groups of

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.

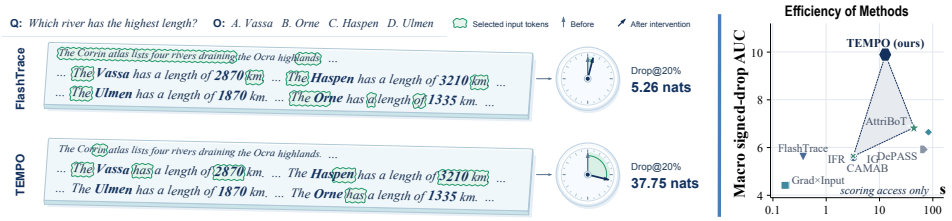

Figure 1: **TEMPO at a glance on Qwen3-8B.** *Left:* source tokens selected by FlashTrace and TEMPO at the same 20% deletion budget on a river-length question. Green outlines mark selected tokens; Drop@20% measures the fall in the fixed answer's log-probability per target token, in nats. *Right:* four-task macro signed-drop AUC versus attribution time per input on a logarithmic time axis; higher is more faithful and farther left is faster.

tokens (Hase et al., 2021). Long horizons also amplify the memory cost of retaining activations, attention maps or pairwise contributions; FlashTrace mitigates this with span-wise aggregation and recursive attribution through intermediate generated text (Pan et al., 2026b), yet its cached attention maps remain quadratic in sequence length, with recomputation trading computation for memory. *Perturbation-based* methods instead probe the target score directly under source interventions and require only scoring access (Covert et al., 2021; Cohen-Wang et al., 2024). Their difficulty is combinatorial: n source units define 2^n masks, so practical methods rely on sampling or polynomial-query approximations such as $O(n)$ leave-one-out scoring and $O(n^2)$ greedy ranking (Covert et al., 2021; Gu et al., 2026), and cheap per-unit estimates miss the interactions that subset search captures at greater query cost (Hase et al., 2021). TRACE offers a route through this trade-off in visual attribution by searching fixed-size binary masks with cross-entropy optimization (Gu et al., 2026), although it still relies on greedy rescoring and Greedy or PhaseWin continuation search to turn the selected set into a full ranking. The open problem is to convert an efficient mask search into a faithful complete order for span-to-span attribution without further model queries.

Inspired by TRACE's mask-search approach (Gu et al., 2026), we propose TEMPO (Text-Enhanced Mask Probability Optimization), a black-box method for long-horizon span-to-span attribution. We assign masking probabilities to source tokens and optimize them using changes in a fixed target span's likelihood under sampled perturbations. Our aim is to convert this probabilistic mask into a competitive attribution ranking without additional model forward passes. We therefore introduce *text-enhanced decoding*, which uses word and clause boundaries as priors for organizing the mask probabilities. The decoder groups neighboring tokens within these boundaries and orders the groups using aggregated mask probabilities, while preserving the best searched set at the search budget. The result is a complete ranking across deletion budgets, obtained entirely from the search output and the source text. Figure 1 illustrates the resulting attribution and its cost on Qwen3-8B. In the river-length example, TEMPO concentrates its selection on the numerical evidence, including the two competing lengths, and at the same 20% deletion budget produces a fixed-answer log-probability drop roughly seven times that of FlashTrace. The lead holds across all four task families, at lower cost than the strongest of them.

To assess faithfulness across diverse textual dependencies, we complement the NIAH retrieval and multi-hop settings studied in prior work (Hsieh et al., 2024; Pan et al., 2026b) with two self-constructed diagnostics: mathematical continuation with injected errors and counterfactual multiple choice. Together, these tasks cover distributed evidence, propagated errors, and isolated values whose modification changes the answer to a question. Reference annotations identify supporting evidence or error sources, enabling a separate evaluation of evidence recovery. On all four tasks, TEMPO outperforms all seven baselines on deletion AUC and leads every macro-average faithfulness metric.

Our experiments also expose a divergence between faithfulness and evidence recovery. Information-flow attribution leads recall on retrieval and multi-hop tasks, whereas leave-one-out attribution leads on math and multiple choice; no evaluated method leads throughout. A BM25 query-matching control that reranks TEMPO's search output raises reference recall by 22% while lowering deletion AUC by 21%. This divergence suggests a challenge in transferring visual mask-search ideas to language: a few decisive tokens may carry a span's influence, so disrupting them can strongly affect target likelihood while recovering little of the annotated span. Faithful attribution therefore requires evaluating behavioral dependence separately from agreement with reference evidence.

Our contributions are as follows.

- We formulate *long-horizon span-to-span attribution* via source interventions with other text fixed, generalizing input attribution and separating source location from attribution granularity.

- We propose TEMPO, combining probabilistic mask optimization with query-free *text-enhanced decoding* to produce budget-nested rankings at a query cost independent of input length and source–target separation.
- On four task families, TEMPO surpasses seven baselines on all macro-average faithfulness metrics, while the most faithful method is not the best localizer.

2 Related Work

Attribution methods. For generated text, ContextCite fits a sparse surrogate to context ablations (Cohen-Wang et al., 2024), AttriBoT accelerates leave-one-out scoring (Liu et al., 2025), and CAMAB searches subsets with Thompson sampling (Pan et al., 2026a). Internal attribution instead tracks information flow (Ferrando & Voita, 2024) or additive components (Hong et al., 2025), with FlashTrace extending tracing to multi-token targets and intermediate generated text (Pan et al., 2026b). Beyond these estimators, optimized masks in vision (Fong & Vedaldi, 2017; Fong et al., 2019; Gu et al., 2026) and learned rationales or sparse search in language (Lei et al., 2016; Hase et al., 2021) cast explanation as subset selection. Hierarchical text explanations further organize words and interactions into spans (Chen et al., 2020; Jin et al., 2020; Tsang et al., 2020). TEMPO connects mask search with this structural perspective: its distinction from TRACE is the query-free conversion of the search output into a complete ranking that preserves the searched set.

Attribution evaluation. Removal-based metrics assess behavioral dependence through comprehensiveness, sufficiency, or insertion/deletion curves (DeYoung et al., 2020; Petsiuk et al., 2018; Chan et al., 2022); context-attribution work uses changes in a fixed response's log-probability (Cohen-Wang et al., 2024; Liu et al., 2025). Their interpretation depends on the intervention operator and the distribution shift induced by perturbation (Hase et al., 2021; Kamahi & Yaghoobzadeh, 2024). Agreement with annotated rationales is a distinct criterion (DeYoung et al., 2020; Jacovi & Goldberg, 2020), and classification studies have shown that it can rank methods differently from behavioral tests (Nguyen, 2018; Atanasova et al., 2020). We follow this distinction by evaluating intervention effects and reference-evidence recovery separately. Appendix F provides the broader research lineage and further evaluation caveats.

3 Method

3.1 Problem setup

Input attribution asks which parts of an input a fixed model output depends on. In its token-level form it ranks input tokens, which in long-context settings may lie far from the output being explained. We treat this setup as a special case of *span-to-span attribution*: one selects source spans and a later target span in a recorded text and explains the target through interventions on the selected source text. The formulation decouples the location of the source from the granularity at which it is attributed.

Source and target selection. Let f be a language model, and split a recorded transcript, with its tokens unchanged, into a prefix x and a continuation y. We select one or more *source spans* in x and a nonempty set T of *target* positions in y; every source precedes every target, and their separation is limited only by the model's context window. The prefix may contain prompt text or earlier generated text, and the target may be a selected output span or the entire continuation. The fixed target is scored by its mean log-probability under teacher forcing,

$$s_f(x, y, T) = \frac{1}{|T|} \sum_{t \in T} \log p_f(y_t \mid x, y_{<t}) \,. \tag{1}$$

No token of y is regenerated, and all text outside the selected source units is held fixed. The selected source text is partitioned into *intervention units* $U = \{u_1, \ldots, u_n\}$ with token costs $c_i = |u_i|$; a set $A \subseteq U$ costs $c(A) = \sum_{u_i \in A} c_i$, and $C = c(U)$ denotes the total. Ordinary input attribution corresponds to taking x as the original prompt and selecting source spans within it; token-level attribution uses single-token units. Taking x to include an earlier part of the generated text yields attribution from that part to a later output span under the same score. A unit is a single token unless the data declare an indivisible atom (e.g. a multi-token numeral); system prompts, chat templates and, where the task fixes them, questions and answer options are never perturbed.

Intervention and attribution output. An *intervention* $I(x, A)$ replaces every unit in A by the padding token, preserving sequence length, positions and attention visibility; its effect on the target is

$$F(A) = s_f(x, y, T) - s_f(I(x, A), y, T)\,, \tag{2}$$

which may be negative and is in general neither monotone nor additive in A. F characterizes the model's behavior under this replacement protocol, and all statements below are relative to it.

An attribution method returns a permutation $\pi = (\pi_1, \ldots, \pi_n)$ of U. Under a budget b, the *strict prefix* $P_\pi(b) = \{\pi_1, \ldots, \pi_k\}$ is the longest initial segment with $c(\{\pi_1, \ldots, \pi_k\}) \le b$; all budgets therefore share one order and induce nested intervention sets. This order is what we optimize and evaluate: score-based methods are ranked by their scores, and set-based methods must specify an order within and beyond the set. A method is *black-box* if it evaluates only Equation (1) on modified inputs, without access to gradients, attention weights or internal states.

Long-horizon attribution. The horizon is the distance between the selected source and target spans in the recorded token sequence. A large separation may arise from a long input, a long generated response, or both: an input passage can be related to a late output span, and an earlier output span to a much later one. Long-context input attribution is thus one instance of the problem; in the general framework both source and target locations vary, subject to their autoregressive order and the context window. In every case the attribution question is the effect of intervening on the selected source units on the selected target, with the intervening text held fixed.

3.2 TEMPO: Text-Enhanced Mask Probability Optimization

TEMPO maps (f, x, y, T, U) to a permutation π in the two stages of Figure 2. The *search* stage (panel A) spends the entire model-query budget on a probabilistic mask over U at a fixed budget $b_0 = \lfloor \rho_0 C \rfloor$ with $\rho_0 = 0.2$, uses only Equation (2), and returns its best set A_0 with the final proposal weights $p \in [0, 1]^n$. The *decoding* stage (panel B) is query-free and uses the word and clause structure of the text to extend the search output to a complete ranking whose b_0-prefix is A_0 (Proposition 3.1).

Search. The search adapts TRACE (Gu et al., 2026), which optimizes a fixed-cardinality mask over superpixels with the cross-entropy method (CEM) (Rubinstein, 1999; de Boer et al., 2005), to text intervention units. Each unit u_i carries a logit θ_i (initially 0), and each round $r = 0, \ldots, R-1$ consists of two steps. *Sample and score.* For each of M candidates j we draw independent noise $g_{j,i} \sim \mathrm{Gumbel}(0, 1)$, form perturbed keys $z_{j,i} = \theta_i^{(r)}/\tau + g_{j,i}$, and traverse units in decreasing $z_{j,i}$, adding each unit that still fits within b_0; this yields the candidate $A_{r,j}$ (Gumbel-top-k sampling (Kool et al., 2019) with unit costs), scored by one forward pass as $F(A_{r,j})$. Because a skipped unit never fits later, every candidate is *maximal feasible*: $c(A_{r,j}) \le b_0 < c(A_{r,j}) + c_i$ for all $u_i \notin A_{r,j}$ (indivisible atoms may leave budget unspent); this ensures that placing the candidate first in the ranking recovers it under the evaluation's strict-prefix rule. *Update.* The top $\lceil \eta M \rceil$ candidates form the elite multiset $\mathcal{E}_r$ with inclusion frequencies $e_i^{(r)} = |\mathcal{E}_r|^{-1} \sum_{j \in \mathcal{E}_r} \mathbf{1}[u_i \in A_{r,j}]$, and each logit takes a damped step toward the logit of its smoothed frequency,

$$\tilde{p}_i^{(r)} = \mathrm{clip}_{[\epsilon, 1-\epsilon]}\big((1-\lambda)\, e_i^{(r)} + \lambda \rho_0\big), \quad \theta_i^{(r+1)} = \mathrm{clip}_{[-L, L]}\big((1-\gamma)\, \theta_i^{(r)} + \gamma \,\mathrm{logit}\, \tilde{p}_i^{(r)}\big), \tag{3}$$

with temperature τ, smoothing λ, step γ and clip bounds ϵ, L (Appendix A.2). We use $R = 5$, $M = 32$ and $\eta = 0.2$. The reported implementation replaces the nominal shrinkage reference ρ_0 by the realized fraction b_0/C (Appendix B.3). The text instantiation differs from TRACE in two respects. (i) The objective is the *necessity* effect F on a fixed generated target, in place of a classifier sufficiency score. (ii) Both search artifacts are retained: the best scored candidate $A_0 \in \arg\max_{r,j} F(A_{r,j})$ and the final proposal weights $p_i = \sigma(\theta_i^{(R)})$. They play complementary roles: A_0 is a verified budget set, whereas p is an empirical proposal distribution shaped by the update rule. Filling b_0 in decreasing p_i need not reproduce A_0, and p enters the decoder only as an ordering signal.

Decoding. The search leaves the order open; the decoder fixes it from text structure alone (Figure 2B) in three steps, requiring only the public text. *Bundle.* Word groups are recovered from Unicode character offsets and clause boundaries from punctuation and line breaks. Walking the text in order, we greedily merge whole words within one clause into a *bundle* of at most 8 tokens; a word or atom longer than 8 tokens is kept whole, so 8 acts as an assembly threshold. *Split.* Each bundle B is split into an inside part $B \cap A_0$ and an outside part $B \setminus A_0$, with empty parts dropped; words may be

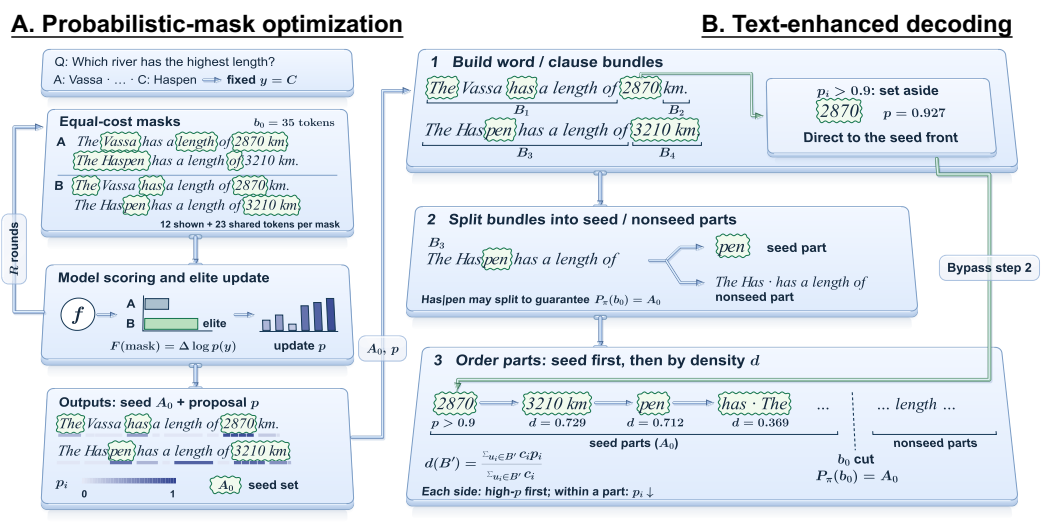

Figure 2: **TEMPO on a multiple-choice input with fixed target $y = C$; boxed units are selected.** *A: probabilistic-mask optimization.* Each round draws masks at budget b_0, scores each by the effect F of Equation (2) in one forward pass, and moves the proposal weights toward the elite candidates, returning the anchor set A_0 and the weights p (shading). *B: text-enhanced decoding*, with no further model queries. (1) Word and clause bundles are read off the text; (2) each bundle is split into $B \cap A_0$ and $B \setminus A_0$; (3) inside parts rank before outside parts, each block by proposal density d, so the b_0 cut recovers A_0 exactly.

split, but units never are, so an atom in A_0 stays whole. *Order.* The ranking π places the inside block before the outside block, and each block is ordered in two passes. The first pass sorts the block by p_i and fixes at its rank every unit with $p_i > 0.9$ (the *saturation rule*): a unit selected by nearly every elite candidate carries evidence that regrouping would dilute. The second pass fills the remaining ranks with bundle parts B' in decreasing proposal density

$$d(B') = \frac{\sum_{u_i \in B'} c_i \, p_i}{\sum_{u_i \in B'} c_i}, \tag{4}$$

ordering units within a part by decreasing p_i and breaking ties by original position; fixed units keep their first-pass rank and still count toward the density of their part. Normalizing by cost makes the ordering cost-aware, and bundling by text structure keeps the units of one expression adjacent.

Proposition 3.1 (Prefix preservation)**.** *Let A_0 be maximal feasible at budget b_0 and let π rank every unit of A_0 before every unit of $U \setminus A_0$. Then $P_\pi(b_0) = A_0$.*

Proof. The first $|A_0|$ units of π are A_0, whose initial segments all cost at most $c(A_0) \le b_0$, so the prefix takes them; the next unit $u_i \notin A_0$ satisfies $c(A_0) + c_i > b_0$ by maximality, so the prefix stops.

Both hypotheses hold by construction, so the decoder leaves the b_0 prefix unchanged; its effect lies in the order inside and beyond A_0 (the b_0 cut in Figure 2B). The density rule is a deterministic heuristic; Appendix C.8 isolates its contribution by ordering the same search output without text structure, and Appendix A reports alternative decoders. The cost of TEMPO is that of the search: decoding issues no forward pass and runs on the CPU in about 9 ms per input for Qwen3-8B.

3.3 Implications for long-horizon span-to-span attribution

TEMPO applies one search and decoding procedure to every choice of source and target in the framework of Section 3.1: U fixes where interventions are allowed and T fixes which output log-probabilities enter the objective, so moving from prompt-to-output attribution to attribution between two generated spans changes only these selections. Each candidate is scored by a single teacher-forced forward pass directly at the target, with the intervening tokens held fixed and their hidden states recomputed, so distant and nearby pairs share the same objective and no intermediate scores or regeneration are needed. The search issues at most RM candidate scores plus one shared reference, a budget independent of input length and source–target separation.

Within a round, candidates are independent given $\theta^{(r)}$ and can be scored in parallel, so with batch size $m \le M$ the search takes at most $R\lceil M/m \rceil$ forward batches plus the reference, and R batches when

all candidates fit in memory; sequence length affects only the cost per batch. The decoder then extends the searched set to a complete order over the source units, preserving A_0 at b_0 by Proposition 3.1 and ordering the rest from text structure without further model queries, so one source–target query yields a budget-nested attribution at no additional cost.

4 EXPERIMENTS

Two questions organize this section, and its subsections answer them in turn: what the method achieves under a shared protocol, and how that achievement relates to agreement with reference evidence. Three ablations then follow in Section 4.4.

RQ1: Does TEMPO achieve higher deletion faithfulness than information-flow, gradient, decomposition and perturbation baselines under one intervention protocol?

RQ2: Does a larger deletion effect coincide with better recovery of reference evidence?

4.1 SETUP

Model and baselines. We run every attribution method on one model, Qwen3-8B (Yang et al., 2025) (bf16, greedy decoding, thinking disabled). We compare seven baselines that span four families: information flow (IFR (Ferrando & Voita, 2024), FlashTrace (Pan et al., 2026b)), gradients (Grad×Input (Ancona et al., 2018), IG (Sundararajan et al., 2017)), decomposition (DePASS (Hong et al., 2025)) and perturbation (AttriBoT (Liu et al., 2025), CAMAB (Pan et al., 2026a)), configured as in Appendix D and Appendix B (baseline implementations and hyper-parameters). **Data.** We use four task families, which probe retrieval (NIAH), multi-span composition (multi-hop QA), reasoning-chain error origin (math, with an injected misleading prefix) and counterfactual dependence (MCQ), each with reference evidence defined by construction (480 inputs from 290 source problems; their sources and construction are in Appendix D). **Metrics.** We measure four metrics: faithfulness, the signed-drop AUC of $F(P_\pi(b))$ over 20 budgets and D@$k\%$; and localization, R@20% and the span-balanced SBα-NDCG@10 against reference evidence (Appendix E, Appendix E.1). All methods share one intervention protocol (Appendix D).

4.2 MAIN RESULT: DELETION FAITHFULNESS (RQ1)

We compare TEMPO with the seven baselines of Section 4.1 on Qwen3-8B across the four task families, all under one intervention protocol. Table 1 carries the deletion-faithfulness side of that comparison, per task and as the four-task macro average, together with attribution time.

TEMPO leads every faithfulness column in each task block and in the macro, which weights each task equally. On every macro metric the strongest baseline is AttriBoT; against it the AUC is 46.1% higher and the fixed-budget drops are higher still. The paired macro AUC difference against AttriBoT is $+3.16$ (95% paired bootstrap CI $[2.75,\ 3.58]$) and the Drop@10/20/30% differences are $+4.36$ $[3.73,\ 5.01]$, $+6.72$ $[6.04,\ 7.44]$ and $+5.58$ $[4.93,\ 6.24]$. Every interval comes from the same procedure, a paired bootstrap that resamples source problems (Appendix D). The other four baselines with a reported interval have a lower AUC than AttriBoT. TEMPO's decoded ordering beats the search-only ordering for all four search seeds (Appendix C.6 lists the per-seed values).

Three further checks support this result. First, TEMPO returns one ordering for the whole deletion range. The same ranking gives the curve at every budget, without further model queries, and the 20% budget the search works at is one point on it. Second, the size of the lead depends on the task, widest on MCQ and narrowest on math, where the spread across methods is 0.19 nats per token and the lead is 0.629 against 0.605 for DePASS (Appendix C.1, Appendix C.3). Third, the lead holds on held-out sources. The extension set removes every input whose source problem was used to select the decoder, and on it the macro AUC is 9.92 against 6.84 for the strongest baseline, a paired difference of $+3.08$ $[2.62,\ 3.56]$ (Appendix C.5). Because TEMPO searches with the same intervention that the evaluation applies, its objective and the metric are aligned by design. The cost is 12.9 s per input (last column of Table 1), 3.4–$6.5\times$ less than the AttriBoT, DePASS and IG implementations it beats (Appendix C.7). Under one intervention protocol, TEMPO therefore achieves higher deletion faithfulness than the information-flow, gradient, decomposition and perturbation baselines, on every task and in the four-task macro.

Table 1: **Deletion faithfulness and attribution time on Qwen3-8B.** Per-task results and the four-task macro average; D@k denotes deletion drop at a $k\%$ budget and time is in seconds; higher AUC and D@k are better and lower time is better; best values in each column are bold.

	NIAH			Multi-hop			Math			MCQ		
Method	AUC	D@20	Time	AUC	D@20	Time	AUC	D@20	Time	AUC	D@20	Time
IFR	6.36	6.74	2.84	11.79	12.71	0.23	0.592	0.462	9.87	3.58	2.10	0.15
FlashTrace	6.18	6.53	0.67	11.91	12.38	0.18	0.597	0.490	0.29	3.77	3.28	0.35
Grad×Input	5.36	4.55	**0.33**	7.07	6.74	**0.11**	0.442	0.266	**0.12**	4.77	4.57	**0.10**
IG	6.98	7.13	198.7	12.63	13.82	38.0	0.592	0.491	38.5	6.34	7.71	57.6
DePASS	7.06	7.48	244.4	12.37	13.51	9.45	0.605	0.541	16.6	3.65	4.10	5.84
AttriBoT	6.65	6.63	162.2	9.08	8.50	6.10	0.571	0.502	5.22	11.08	14.45	3.85
CAMAB	6.59	6.72	6.06	11.57	12.53	2.16	0.538	0.399	2.47	3.87	2.62	2.15
TEMPO	**8.00**	**9.59**	24.4	**14.04**	**18.52**	8.68	**0.629**	**0.639**	9.91	**17.33**	**28.23**	8.56

	Four-task macro									
Method	AUC	D@10	D@20	D@30	Time	Method	AUC	D@10	D@20	D@30	Time
IFR	5.58	5.44	5.50	5.69	3.27	DePASS	5.92	6.55	6.41	6.19	69.1
FlashTrace	5.61	5.54	5.67	6.00	0.37	AttriBoT	6.84	7.81	7.52	7.50	44.3
Grad×Input	4.41	3.40	4.03	4.35	**0.17**	CAMAB	5.64	4.58	5.57	5.68	3.21
IG	6.64	7.00	7.29	7.45	83.2	**TEMPO**	**10.00**	**12.17**	**14.24**	**13.08**	12.9

4.3 Faithfulness and localization diverge (RQ2)

We next ask whether the same rankings recover the annotated evidence. Table 2 gives this comparison for the three methods we discuss (FlashTrace, AttriBoT and TEMPO); Table 8 gives all eight methods with the remaining budgets.

Table 2: **Agreement with reference annotations on Qwen3-8B.**

	Recall@20%↑				SBα-NDCG@10↑			
Method	NIAH	Multi-hop	Math	MCQ	NIAH	Multi-hop	Math	MCQ
FlashTrace	0.849	0.488	0.326	0.027	0.999	0.533	0.175	0.000
AttriBoT	0.543	0.452	**0.375**	**0.722**	0.998	**0.593**	**0.256**	**0.672**
TEMPO	0.273	0.249	0.254	0.656	0.469	0.342	0.189	0.656

TEMPO does not lead here. The leading method depends on the task, and NDCG and recall disagree with each other. On NIAH the information-flow methods recover the needle almost completely while TEMPO does not; on multi-hop IFR leads recall and AttriBoT leads NDCG; on math and MCQ AttriBoT leads both, with TEMPO second on MCQ (Appendix C.1, Appendix C.4). The most faithful method is not the best localizer, and the best-localizing method changes with the task, with the math effects an order of magnitude smaller than the rest (Table 8).

Recall scores coverage of the annotated evidence, not the tokens the answer depends on. On NIAH the same question is paired with a short and a long needle, so only the size of the annotated evidence changes. The faithfulness ranking is the same for both, while recall falls for every method as the needle grows, so no method leads recall consistently (Appendix C.2).

Information-flow scores follow the route a token feeds, so tokens of the same kind score alike; this prior matches whole-sentence annotations, and FlashTrace is ahead of TEMPO on the NIAH needle and multi-hop support sentences while its recall collapses on math and MCQ (Table 2). Where the answer turns on a single edited value, one span at a time is the right unit. AttriBoT, whose leave-one-out scoring isolates single spans, leads recall on math and MCQ, whereas ours is token-padding leave-one-out (Appendix B, which contrasts the two units). Annotation type decides the leader. Whole-sentence and dispersed-support annotations on NIAH and multi-hop favor information-flow scores; key-value and local-error annotations on math and MCQ favor span-level leave-one-out.

The evidence a response depends on is discrete, so flipping the response requires breaking a few tokens rather than removing many. Accordingly, spreading the same deletion mass over several key

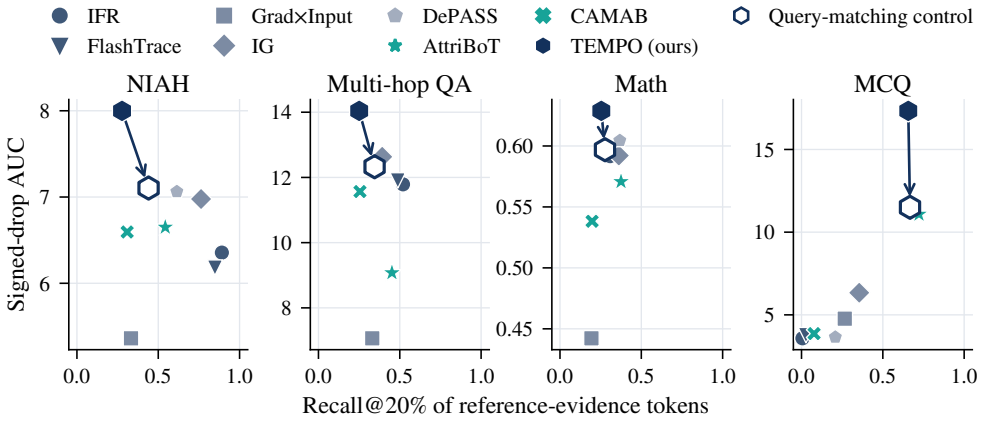

Figure 3: **Faithfulness against localization per task on Qwen3-8B.**

sentences moves the decision more than concentrating it on fewer, which is what the bundle-density ordering of Section 3.2 does (Table 16, Section 4.4).

We next ask how the two metrics respond when only the decoder changes. We keep TEMPO's search output and replace its ordering rule with query matching. With the question as query and the perturbable sentences as documents, the proposal weights are re-ranked by the rule of Equation (5). The re-ranking rule replaces the proposal weights by

$$p_i' = p_i + 0.3 \, \max\big(\max_j p_j - \min_j p_j, \, 0.05\big) \, h_i, \tag{5}$$

with BM25 parameters $k_1 = 1.2$ and $b = 0.75$ (Robertson & Zaragoza, 2009), sentence scores normalized to $h_i \in [0, 1]$, and the anchor-set constraint released so that the b_0-prefix may change. The control uses fixed parameters for every task, sees no annotations and queries the model no further. It serves as an analysis instrument.

Figure 3 plots faithfulness against localization per task for every method, with the arrow from TEMPO to the control, and Table 17 tabulates the plotted coordinates. Re-ranking toward sentences that match the question lexically raises Recall@20% by 22% and SBα-NDCG@10 by 36%, and lowers the AUC by 21% and Drop@20% by 10%, every reported interval excluding zero (Table 3, Figure 3). The decoder of the control optimizes lexical overlap between a unit and the question, so it moves the deleted set toward question-matching sentences and spreads it across more sentences, which raises agreement with reference evidence and lowers the deletion effect. The two objectives therefore pull the same search output in opposite directions.

Table 3: **Query-matching control against TEMPO of four-task macro.**

Metric	TEMPO	Query-matching control	Δ [95% CI]
AUC↑	9.99	7.88	−2.12 [−2.39, −1.86]
Drop@20%↑	14.24	12.80	−1.44 [−1.64, −1.24]
Recall@20%↑	0.360	0.439	+0.080 [+0.068, +0.092]
SBα-NDCG@10↑	0.420	0.571	+0.151

Together, the two parts answer RQ2 negatively. A larger deletion effect does not coincide with better recovery of reference evidence on these tasks, since no method is uniformly best against the annotations and re-ranking the same search output toward question-matching sentences raises that agreement while lowering the deletion effect. The two evaluation objectives therefore diverge.

4.4 Ablations: decoder ordering, model and target type

Ablation 1: decoder ordering. The reported time covers both search and decoding, and the decoder acts only on the order in which units are deleted. Because it leaves the b_0-prefix at A_0 (Section 3.2), ordering the same search output without text structure changes only the curve between budgets.

The macro AUC falls from 10.00 to 9.00 (paired difference +1.00 [0.81, 1.20], per-task values in Table 16), with identical Drop@20% and Recall@20% on every input and the gain largest on MCQ and smallest on math. At the other budgets the decoder is ahead on Qwen3-8B on every task except math at 30%, where the two are within 0.003. Figure 4 plots the signed drop of the search-only ordering, TEMPO and the query-matching control of Section 4.3 at the 10%, 20% and 30% budgets on Qwen3-8B. The search-only ordering and TEMPO coincide at 20% by construction, whereas the control releases the anchor set and so changes the 20% set.

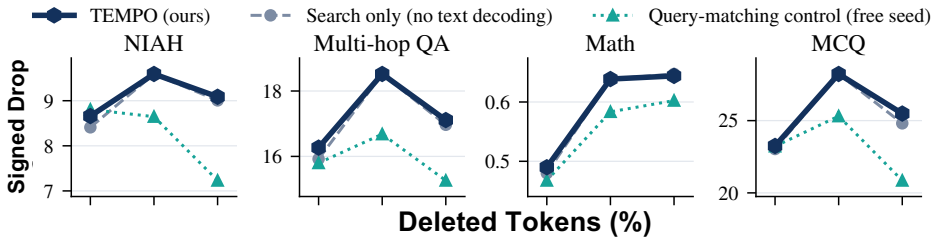

Figure 4: **Comparison of different decoding designs on Qwen3-8B.**

Ablation 2: the model. Qwen3-8B supports all four baseline families; to ask whether the advantage survives another architecture, we repeat the comparison on Qwen3.5-9B, which interleaves 24 gated delta-rule layers with 8 full-attention layers. Only model-agnostic methods can run there, because the information-flow and decomposition baselines are defined over softmax attention (Section 2) and IG exceeds the memory available at these input lengths. The comparison therefore covers the methods in Table 4 on the same NIAH, multi-hop and math inputs, with frozen hyper-parameters and the protocol of Section 4.1 (Appendix B gives the configurations).

Results are a three-task macro over the 336 faithfulness inputs of NIAH, multi-hop and math, not numerically comparable to the four-task macro of Table 1 (Table 4). TEMPO keeps its lead over the three model-agnostic baselines in every column of Table 4, with the largest margins at the fixed budgets, so the faithfulness advantage of Table 1 survives the change of architecture.

Table 4: **Three-task macro on Qwen3.5-9B.**

Method	AUC↑	D@10%↑	D@20%↑	D@30%↑	R@20%↑	NDCG@10↑	Time (s)↓
Grad×Input (Ancona et al., 2018)	2.79	1.44	1.84	2.24	0.292	0.261	**0.80**
AttriBoT (Liu et al., 2025)	3.22	2.74	2.77	2.97	**0.567**	**0.644**	88.4
CAMAB (Pan et al., 2026a)	3.66	2.99	3.36	3.64	0.287	0.316	6.67
TEMPO	**3.90**	3.87	**4.30**	**4.08**	0.293	0.356	26.8

Ablation 3: the target type. The main experiments attribute the tokens of a generated response; here the target is a whole generated program instead. We attribute complete generated programs on Exercism Python tasks, using the pass-filtered single-generation protocol of FlashTrace (Pan et al., 2026b) on our own exercise pool. A single greedy generation per task passed all reference tests for 48 of 140 exercises, and those whole programs are attributed with the entire generated program as the target. There is no reference evidence for code, so only faithfulness is reported (Table 5). These 48 tasks are the pass-filtered subset of the 140-exercise pool, and TEMPO has the highest AUC on them, 0.713 against 0.684 for FlashTrace, with the widest margins at fixed budgets (Drop@20% 0.891 vs. 0.568 for the best baseline). With 48 inputs the AUC interval against FlashTrace includes zero and that against AttriBoT does not; the intervals themselves are in Appendix C.9.

Table 5: **Code generation on Qwen3-8B.**

Method	AUC↑	Drop@10%↑	Drop@20%↑	Drop@30%↑	Time (s)↓
IFR (Ferrando & Voita, 2024)	0.681	0.454	0.568	0.616	11.5
FlashTrace (Pan et al., 2026b)	0.684	0.451	0.543	0.626	0.29
Grad×Input (Ancona et al., 2018)	0.371	0.173	0.225	0.303	0.29
DePASS (Hong et al., 2025)	0.609	0.427	0.484	0.527	46.8
AttriBoT (Liu et al., 2025)	0.516	0.433	0.464	0.492	27.8
CAMAB (Pan et al., 2026a)	0.597	0.327	0.474	0.577	2.96
TEMPO	**0.713**	**0.686**	**0.891**	**0.874**	11.9

5 Conclusion

We introduced TEMPO for long-horizon span-to-span attribution, measuring how interventions on earlier text affect a fixed target while holding the remaining transcript unchanged. By combining probabilistic mask search with text-enhanced decoding, TEMPO turns a search at one budget into a complete ranking without additional model queries, preserving the best searched set at that budget. Across four task families on Qwen3-8B, it outperforms seven baselines on all macro-average faithfulness metrics. However, a query-matching control improves evidence recall while weakening deletion effects, showing that behavioral dependence and annotation agreement are distinct evaluation objectives. Together, these findings support using text structure to extend a limited intervention search and call for exploring better attribution targets and intervention methods.

AI USE STATEMENT

We have not used generative AI tools for any of the tasks with required disclosure, and we used them for the tasks with recommended disclosure: to edit this paper for readability, to suggest its structure, to format references, and to check its internal numerical and terminological consistency against the released summary data and the plotting scripts. We have reviewed all AI-assisted work: every edit was checked against the original text, and every numerical claim was checked against the source data. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We give the attribution protocol, the metric definitions and the implementation details, including baseline configurations and the hyper-parameters of TEMPO, in Appendix D, Appendix E and Appendix B. Per-task results, the held-out extension set, the search-seed sweep and per-task timings are in Appendix C.1, Appendix C.5, Appendix C.6 and Appendix C.7. The summary data behind the main results table, the per-task table, the needle split, the decoder ablation, the Qwen3.5-9B macro and the control table, together with the plotting and consistency-checking scripts used to verify those numbers and to draw the figures from them, are included in the supplementary material. The four task families and their per-input annotations are released as ATTR-BENCH. The code-generation, extension-set, seed-stability, Llama timing and MCQ-variant tables report further measurements that are not accompanied by their raw outputs.

ETHICS STATEMENT

This work attributes fixed model responses over public models and public benchmark data, and involves no human subjects. Attribution that identifies which part of a prompt a response depended on also identifies which part of a prompt can be used to steer a model, which is the security question we raise in Section 1.

REFERENCES

Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. In *ACL*, 2020.

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 ICML, PMLR 235*, 2024. arXiv:2402.05602.

Kirill Afendulev, Alexey Dontsov, Elena Tutubalina, and Anton Korznikov. What attention recalls and recurrence controls in hybrid language models. *arXiv preprint arXiv:2609.04434*, 2026.

Ameen Ali, Itamar Zimerman, and Lior Wolf. The hidden attention of Mamba models. *arXiv preprint arXiv:2403.01590*, 2024.

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *ICLR*, 2018.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. A diagnostic study of explainability techniques for text classification. In *EMNLP*, pp. 3256–3274, 2020.

Somnath Banerjee, Pranav Jha, Rima Hazra, and Animesh Mukherjee. Lost in interpretation: The plausibility-faithfulness trade-off in cross-lingual explanations. *arXiv preprint arXiv:2605.19274*, 2026.

Oren Barkan, Edan Hauon, Avi Caciularu, Ori Katz, Itzik Malkiel, Omri Armstrong, and Noam Koenigstein. Grad-SAM: Explaining transformers via gradient self-attention maps. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 2882–2887, 2021.

Jasmijn Bastings and Katja Filippova. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pp. 149–155, 2020.

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. Interpretable neural predictions with differentiable binary variables. In *Proceedings of ACL*, 2019. arXiv:1905.08160.

Abhinaba Basu and Pavan Chakraborty. ICE: Intervention-consistent explanation evaluation with statistical grounding for LLMs. *arXiv preprint arXiv:2603.18579*, 2026.

Hector Borobia, Elies Seguí-Mas, and Guillermina Tormo-Carbó. Component ablation for efficient hybrid language model architectures: Performance, resilience, and compression implications. *arXiv preprint arXiv:2603.22473*, 2026.

Stephanie Brandl and Oliver Eberle. A systematic comparison between extractive self-explanations and human rationales in text classification. In *Proceedings of the 6th Workshop on Trustworthy NLP (TrustNLP 2026)*, pp. 563–583, San Diego, California, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.trustnlp-main.44.

Chun Sik Chan, Huanqi Kong, and Liang Guanqing. A comparative study of faithfulness metrics for model interpretability methods. In *ACL*, pp. 5029–5038, 2022.

Hanjie Chen, Guangtao Zheng, and Yangfeng Ji. Generating hierarchical explanations on text classification via feature interaction detection. In *Proceedings of ACL*, 2020. arXiv:2004.02015.

Ruoyu Chen, Hua Zhang, Siyuan Liang, Jingzhi Li, and Xiaochun Cao. Less is more: Fewer interpretable region via submodular subset selection. In *ICLR*, 2024.

Ruoyu Chen, Siyuan Liang, Jingzhi Li, Shiming Liu, Maosen Li, Zhen Huang, Hua Zhang, and Xiaochun Cao. Interpreting Object-level Foundation Models via Visual Precision Search. In *CVPR*, 2025.

Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. ContextCite: Attributing model generation to context. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2409.00729.

Ian Covert, Scott Lundberg, and Su-In Lee. Explaining by removing: A unified framework for model explanation. *Journal of Machine Learning Research*, 22, 2021. arXiv:2011.14878.

Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134:19–67, 2005.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. ERASER: A benchmark to evaluate rationalized NLP models. In *ACL*, pp. 4443–4458, 2020.

Joakim Edin, Andreas Geert Motzfeldt, Casper L. Christensen, Tuukka Ruotsalo, Lars Maaløe, and Maria Maistro. Normalized AOPC: Fixing misleading faithfulness metrics for feature attribution explainability. In *Proceedings of ACL*, 2025. arXiv:2408.08137.

James Enouen, Hootan Nakhost, Sayna Ebrahimi, Sercan O. Arik, Yan Liu, and Tomas Pfister. TextGenSHAP: Scalable post-hoc explanations in text generation with long documents. *arXiv preprint arXiv:2312.01279*, 2023.

Javier Ferrando and Elena Voita. Information flow routes: Automatically interpreting language models at scale. In *EMNLP*, 2024.

Javier Ferrando, Gerard I Gállego, and Marta R Costa-jussà. Measuring the mixing of contextual information in the transformer. In *EMNLP*, pp. 8698–8714, 2022.

Ruth Fong, Mandela Patrick, and Andrea Vedaldi. Understanding deep networks via extremal perturbations and smooth masks. In *Proceedings of ICCV*, 2019. arXiv:1910.08485.

Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of ICCV*, 2017. arXiv:1704.03296.

Zihan Gu, Ruoyu Chen, Junchi Zhang, Yue Hu, Hua Zhang, and Xiaochun Cao. PhaseWin search framework enable efficient object-level interpretation. *arXiv preprint arXiv:2511.10914*, 2025.

Zihan Gu, Jiayu Wang, Hua Zhang, and Yue Hu. A good initialization is all you need for faithful visual attribution. *arXiv preprint arXiv:2607.06726*, 2026.

Peter Hase, Harry Xie, and Mohit Bansal. The out-of-distribution problem in explainability and search methods for feature importance explanations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. arXiv:2106.00786.

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 Datasets and Benchmarks Track*, 2021b.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Xiangyu Hong, Che Jiang, Kai Tian, Biqing Qi, Youbang Sun, Ning Ding, and Bowen Zhou. DePass: Unified feature attributing by simple decomposed forward pass. In *NeurIPS*, 2025.

Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. In *NeurIPS*, pp. 9737–9748, 2019.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *Conference on Language Modeling (COLM)*, 2024.

Alon Jacovi and Yoav Goldberg. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *ACL*, pp. 4198–4205, 2020.

Sarthak Jain and Byron C Wallace. Attention is not explanation. In *NAACL*, pp. 3543–3556, 2019.

Xisen Jin, Zhongyu Wei, Junyi Du, Xiangyang Xue, and Xiang Ren. Towards hierarchical importance attribution: Explaining compositional semantics for neural sequence models. In *International Conference on Learning Representations (ICLR)*, 2020. arXiv:1911.06194.

Yiming Ju, Yuanzhe Zhang, Zhao Yang, Zhongtao Jiang, Kang Liu, and Jun Zhao. Logic traps in evaluating attribution scores. In *ACL*, pp. 5911–5922, 2022.

Sepehr Kamahi and Yadollah Yaghoobzadeh. Counterfactuals as a means for evaluating faithfulness of attribution methods in autoregressive language models. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 452–468, 2024.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. Attention is not only a weight: Analyzing transformers with vector norms. In *EMNLP*, pp. 7057–7075, 2020.

Wouter Kool, Herke van Hoof, and Max Welling. Stochastic beams and where to find them: The Gumbel-top-k trick for sampling sequences without replacement. In *Proceedings of ICML*, 2019. arXiv:1903.06059.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, et al. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*, 2023.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. Rationalizing neural predictions. In *Proceedings of EMNLP*, 2016. arXiv:1606.04155.

Fengyuan Liu, Nikhil Kandpal, and Colin Raffel. AttriBoT: A bag of tricks for efficiently approximating leave-one-out context attribution. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2411.15102.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NeurIPS*, pp. 4765–4774, 2017.

Qing Lyu, Marianna Apidianaki, and Chris Callison-Burch. Towards faithful model explanation in NLP: A survey. *Computational Linguistics*, 50(2):657–723, 2024.

Andreas Madsen, Nicholas Meade, Vaibhav Adlakha, and Siva Reddy. Evaluating the faithfulness of importance measures in NLP by recursively masking allegedly important tokens and retraining. In *Findings of EMNLP*, 2022a. arXiv:2110.08412.

Andreas Madsen, Siva Reddy, and Sarath Chandar. Post-hoc interpretability for neural NLP: A survey. *ACM Comput. Surv.*, 55(8):1–42, 2022b.

Dong Nguyen. Comparing automatic and human evaluation of local explanations for text classification. In *NAACL*, pp. 1069–1078, 2018.

Lucas Monteiro Paes, Dennis Wei, Hyo Jin Do, et al. Multi-level explanations for generative language models. In *Proceedings of ACL*, 2025. arXiv:2403.14459.

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, 2026a.

Wenbo Pan, Zhichao Liu, Xianlong Wang, Haining Yu, and Xiaohua Jia. Towards long-horizon interpretability: Efficient and faithful multi-token attribution for reasoning LLMs. *arXiv preprint arXiv:2602.01914*, 2026b. ICML 2026.

Vitali Petsiuk, Abir Das, and Kate Saenko. RISE: Randomized input sampling for explanation of black-box models. In *BMVC*, pp. 151, 2018.

Jirui Qi, Gabriele Sarti, Raquel Fernández, and Arianna Bisazza. Model internals-based answer attribution for trustworthy retrieval-augmented generation. In *Proceedings of EMNLP*, 2024. arXiv:2406.13663.

Farnoush Rezaei Jafari, Grégoire Montavon, Klaus-Robert Müller, and Oliver Eberle. MambaLRP: Explaining selective state space sequence models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2406.07592.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.

Reuven Y. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 1:127–190, 1999.

Lloyd S Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2, 1953.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, pp. 3319–3328, 2017.

Quoc-Huy Trinh, Lin Zhu, and Sebastian Szyller. How context attribution handles what the model already knows. *arXiv preprint arXiv:2607.23804*, 2026.

Michael Tsang, Sirisha Rambhatla, and Yan Liu. How does this interaction affect me? interpretable attribution for feature interactions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. arXiv:2006.10965.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2305.04388.

Chase Walker, Dominic Simon, Kenny Chen, and Rickard Ewetz. Attribution quality metrics with magnitude alignment. In *Proceedings of IJCAI*, 2024.

Yanting Wang, Wei Zou, Runpeng Geng, and Jinyuan Jia. TracLLM: A generic framework for attributing long context LLMs. In *USENIX Security Symposium*, 2025. arXiv:2506.04202.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *EMNLP*, pp. 11–20, 2019.

Yingtai Xiao, Yuqing Zhu, Sirat Samyoun, Wanrong Zhang, Jiachen T Wang, and Jian Du. TokenShapley: Token level context attribution with Shapley value. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 3882–3894, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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, 2018.

Chenyang Zhao, Kun Wang, Xingyu Zeng, Rui Zhao, and Antoni B Chan. Gradient-based visual explanation for transformer-based CLIP. In *ICML*, pp. 61072–61091, 2024.

Zhixue Zhao and Nikolaos Aletras. Incorporating attribution importance for improving faithfulness metrics. In *ACL*, pp. 4732–4745, 2023.

Zhixue Zhao and Boxuan Shan. ReAGent: A model-agnostic feature attribution method for generative language models. *arXiv preprint arXiv:2402.00794*, 2024.

A Method Details and the Role of Linguistic Priors

This appendix specifies the search and decoding procedures of Section 3 together with their implementation conventions. It covers the two anchor-preserving decoders, the probability ordering used by the search-only ablation and the text-enhanced ordering used by TEMPO, and the BM25 query-matching control, which reuses the same search output and releases the anchor constraint. It closes with a discussion of when a model whose dependencies agree closely with human linguistic structure could make explicit linguistic decoding redundant, separating properties of the algorithms from a hypothesis about model alignment.

A.1 Scoring contract and intervention domain

An attribution instance fixes (f, x, y, T, U) before search. The source units are nonempty, disjoint sets of token positions with positive integer costs $c_i = |u_i|$, ordered by their first position, and every source position precedes every target position. When the boundary between x and y is moved to select an earlier generated source span, the original token sequence, chat delimiters, position identifiers and causal attention convention are preserved; the text is never re-rendered through a different chat template. Protected text remains in the model context and outside U.

The scorer replaces exactly the tokens belonging to the selected units, retains their attention visibility, and evaluates the same recorded target under Equation (1). Padding replacement is an intervention on token identities; the batch padding and attention mask are unchanged. Log-probabilities are read at the prediction position preceding each target token, with the shift applied once, and the target denominator $|T|$ is fixed across candidates. Replacing the mean by the sum within one search multiplies every $F(A)$ by the same positive constant and preserves elite selection in exact arithmetic; reported effects use the mean. Signed effects, including negative ones, are retained.

The reference score is computed once per instance. A cache maps a canonical tuple of selected unit indices to its score and is valid for one fixed model, transcript, target and intervention operator. Scoring uses evaluation mode and a fixed numerical configuration, so a repeated mask reuses its score. Non-finite scores are treated as errors. Batching preserves each example's positions, attention mask and target positions, and replay checks compare batched with individual scores before the two are treated as interchangeable. The same contract applies at every source–target separation.

A.2 BUDGETED CROSS-ENTROPY SEARCH

A.2.1 PROPOSAL DISTRIBUTION AND COST-CONSTRAINED SAMPLING

Let $b_0 = \lfloor \rho_0 C \rfloor$, with $C = \sum_i c_i$ and $\rho_0 = 0.2$. The search samples sets from the maximal feasible family

$$\mathcal{M}_{b_0} = \{A \subseteq U : c(A) \le b_0, \quad c(A) + c_i > b_0 \text{ for every } u_i \notin A\}. \tag{6}$$

Maximality refers to the budget: no further unit can be added. Since F is nonmonotone, the best set in this family may differ from the best set among all sets of cost at most b_0; the restriction concentrates evaluations at the searched budget.

Each unit has a proposal logit θ_i, initially zero. For each candidate, independent Gumbel noise is added to θ_i/τ and all units are sorted by the perturbed keys. Before cost packing, the resulting permutation ω follows the without-replacement ranking law of Gumbel sampling (Kool et al., 2019):

$$q_\theta(\omega) = \prod_{k=1}^{n} \frac{\exp(\theta_{\omega_k}/\tau)}{\sum_{\ell=k}^{n} \exp(\theta_{\omega_\ell}/\tau)}. \tag{7}$$

The candidate is obtained by traversing ω and accepting each unit that fits the remaining budget. For equal unit costs this is Gumbel-top-k selection with $k = \lfloor b_0/c_i \rfloor$; with unequal costs it is a deterministic packing of a random order. The resulting law differs from independent Bernoulli sampling and from a uniform draw over feasible sets, and independent candidates can coincide even though each contains no repeated unit.

Packing and evaluation treat a unit that fails to fit differently: packing skips it and continues, whereas the strict prefix of the evaluation stops at it. For costs $(2, 3, 1)$ in priority order at budget 3, packing returns the first and third units, while a strict prefix of the same order contains only the first. Once the packed set is placed before all remaining units, its maximality guarantees that the strict prefix recovers it (Proposition 3.1). An indivisible atom is never split to fill residual budget.

A.2.2 ELITE FITTING AND THE MEANING OF THE UPDATE

In each round, let $K = \lceil \eta M \rceil$ and retain the K highest-scoring sampled candidates with multiplicity; for $M = 32$ and $\eta = 0.2$, $K = 7$. Elite ties are resolved by sampling order, and ties for the best candidate keep the earliest observed winner. The inclusion frequency e_i counts how often unit i occurs among the K samples, so deduplicating expensive evaluations leaves the elite multiset intact.

The frequency update has the standard interpretation of fitting a product-Bernoulli surrogate to the elite masks in CEM (Rubinstein, 1999; de Boer et al., 2005). With $a_{j,i} = \mathbf{1}[u_i \in A_j]$, the unregularized coordinate fit is

$$\widehat{q}_i \in \arg\max_{q \in [0,1]} \sum_{j \in \mathcal{E}} \left[a_{j,i} \log q + (1 - a_{j,i}) \log(1 - q) \right], \qquad \widehat{q}_i = e_i. \tag{8}$$

Mixing the mean elite log-likelihood with a Bernoulli reference of mean ρ_0, with weights $1 - \lambda$ and λ, gives the unconstrained maximizer $(1 - \lambda)e_i + \lambda\rho_0$. Equation (3) clips this value away from 0 and 1, moves the old logit toward its logit with step γ, and clips the result to $[-L, L]$. Probability shrinkage, logit damping and logit clipping are applied in that order; damping in logit space differs from averaging probabilities.

This derivation gives the update its surrogate interpretation. An exact maximum-likelihood fit of the packed-set distribution induced by Equation (7) would be a different object, since the cost constraint couples unit inclusion and packing maps permutations to sets many-to-one. Accordingly, $p_i = \sigma(\theta_i^{(R)})$ is a proposal weight: at initialization all p_i equal $1/2$ although equal-cost candidates select about 20% of the units, and neither $\sum_i p_i$ nor $\sum_i c_i p_i$ is constrained to equal b_0. Adding a common constant to every logit leaves the sampling-order distribution unchanged before clipping while changing every sigmoid weight, so absolute sigmoid values are interpretable only under the declared update convention.

Shrinkage is toward the nominal fraction ρ_0. Shrinking toward the realized ratio b_0/C is a nearby convention that differs by less than $1/C$ when rounding is active. The reported experiments use this realized-ratio convention, as specified in Appendix B.3; the two conventions must not be silently interchanged. Likewise, p is read from the final update and is never averaged across rounds.

Algorithm 1: Budgeted CEM search for a fixed attribution instance

Input: U, costs c_i, budget b_0, fixed score F, $R, M, \eta, \tau, \lambda, \gamma, \epsilon, L$
Output: Best observed set A_0 and final proposal weights p
Initialize $\theta \leftarrow 0$, an instance-local score cache, and no incumbent;
for $r = 0, \ldots, R-1$ **do**
  for $j = 1, \ldots, M$ **do**
    Draw independent $g_{j,i} \sim \mathrm{Gumbel}(0,1)$ and sort $\theta_i/\tau + g_{j,i}$ decreasingly;
    Set $A_{r,j} \leftarrow \varnothing$; traverse this order, adding each unit whose cost still fits b_0;
  end
  Batch-score all previously unseen masks; retrieve cached scores for repeats;
  Update A_0 if a candidate strictly improves the incumbent score;
  Select the top $K = \lceil \eta M \rceil$ sample indices, retaining repeated masks;
  Compute $e_i = K^{-1} \sum_{j \in \mathcal{E}_r} \mathbf{1}[u_i \in A_{r,j}]$;
  Apply probability shrinkage, clipping, damped logit update and logit clipping as in Equation (3);
end
Return A_0 and $p_i = \sigma(\theta_i)$;

A.2.3 Parameters, edge cases and cost

The settings are $R = 5$, $M = 32$, $\eta = 0.2$, $\tau = 1$, $\lambda = 0.05$, $\gamma = 0.7$, $\epsilon = 10^{-4}$ and $L = 8$. The temperature sets the relative strength of proposal logits and sampling noise and is unrelated to generation temperature. Shrinkage and finite logits maintain exploration; discovery of a global optimum within five rounds is not guaranteed. The best observed score is nondecreasing as candidates accumulate, whereas the quality of a new round's samples need not improve monotonically, which is why both A_0 and p are retained: the best candidate may have appeared before the last round.

If U is empty, the attribution is the empty permutation. If no unit fits b_0, all candidates are empty, $F(\varnothing) = 0$, the order is uninformative at the searched budget, and ties are resolved deterministically; atoms are still never split and the budget is unchanged. For nondegenerate instances, at most RM distinct candidate evaluations and one reference are needed, often fewer because of caching. With batch size $m \leq M$ there are at most $R\lceil M/m \rceil$ candidate-scoring batches, and the rounds remain sequential because each proposal depends on the current elite set. Unit sorting costs $O(RMn \log n)$ in a direct implementation, in addition to model scoring, whose per-batch cost grows with sequence length. A fixed query count therefore bounds the number of scoring calls, while computation, memory and ranking quality still depend on the size of the attribution problem.

A.3 Two anchor-preserving decoders

Both decoders consume the same (U, c, A_0, p), use the same tie-breaking order, return a permutation of exactly U, and issue no further search, model queries or use of reference-evidence labels. Let $Q_{\mathrm{in}} = A_0$ and $Q_{\mathrm{out}} = U \setminus A_0$. Each decoder orders the two blocks separately and places the inside block before the outside block, so both satisfy $P_\pi(b_0) = A_0$ by Proposition 3.1, including when atoms leave budget unspent. Their agreement on Drop@20% and Recall@20% is therefore structural and says nothing about the remainder of the ranking.

A.3.1 Probability ordering: the search-only decoder

Within each block, units are sorted by decreasing p_i with ties broken by original token position:

$$\pi_{\mathrm{prob}} = \mathrm{sort}_{(-p_i,\, \mathrm{pos}(u_i))}(A_0) \,\|\, \mathrm{sort}_{(-p_i,\, \mathrm{pos}(u_i))}(U \setminus A_0), \tag{9}$$

where $\|$ denotes concatenation. This is the search-only ordering of Appendix C.8: it removes the linguistic decoder while preserving the searched intervention. A global sort of all units by p_i would be a different method, since it could change the b_0 set. Sorting logits in place of sigmoid weights gives the same order in exact arithmetic, provided clipping and ties are handled consistently.

The decoder assumes finite proposal weights aligned one-to-one with the units, a maximal feasible anchor and stable ties. Its modeling assumption is that, within each block, higher proposal weight is useful evidence about which units to place earlier at budgets other than b_0. CEM optimizes sets at one budget and leaves this assumption open: co-selected units can obtain similar weights even when their individual effects differ, and interchangeable units can divide their elite frequency. The decoder is thus a deterministic extension of the search whose prefixes beyond b_0 carry no optimality guarantee.

Algorithm 2: Text-enhanced decoding of fixed search artifacts

Input: Units U, costs c_i, anchor A_0, weights p_i, source offsets and text
Output: A permutation π_{text} preserving the anchor prefix
Build a disjoint bundle partition using word groups, clause boundaries and the eight-token threshold;
foreach Q *in* $(A_0, U \setminus A_0)$ **do**
  Compute the probability order v of Q;
  Reserve in v each rank occupied by a unit with $p_i > 0.9$;
  Form all nonempty bundle parts $B' = B \cap Q$;
  Compute $d(B')$ from all its units, including reserved ones;
  Sort parts by decreasing $d(B')$ and units within parts by decreasing p_i, with positional ties;
  Concatenate the sorted parts and remove reserved units to obtain a list w;
  Fill the unreserved ranks of v from w in order, retaining every reserved rank;
end
Concatenate the inside-block and outside-block orders to obtain π_{text};

A.3.2 TEXT-ENHANCED ORDERING: BUNDLES, SPLIT AND SATURATION

The text-enhanced decoder adds a local structural prior to this extension. It assumes that units of a short linguistic expression often carry related evidence, so keeping them adjacent in the ranking is useful when finite search has separated their proposal weights. The assumption concerns ordering between budgets only; CEM masks remain free to delete parts of words or clauses.

Segmentation is computed from the original public source text and token-to-character offsets, before any padding replacement, with character and byte positions converted rather than interchanged. Lexical grouping recognizes words, numerals and punctuation, and clause boundaries are surface cues from punctuation and line breaks. A token or indivisible unit that crosses lexical groups joins those groups for grouping purposes, so each unit is assigned once; distinct token positions with overlapping Unicode offsets remain distinct tokens for costing. An atom takes precedence over a textual boundary and is assigned to a clause by a fixed convention, its first character. For disjoint source regions, grouping is performed within each region, so selecting distant spans never merges them into one expression. In source order, whole lexical groups within a clause are merged until the next group would exceed eight tokens; a new bundle starts there or at a clause boundary, and a word or atom longer than eight tokens stays whole. Eight thus acts as an assembly threshold, and the bundles partition the intervention units. The segmentation convention is part of the decoder specification, since a punctuation heuristic is not equally suited to every language, domain or source layout.

Every bundle B is then split into $B \cap A_0$ and $B \setminus A_0$, discarding empty parts; the split may separate a word's units between the anchor blocks but never splits a unit. Within a block Q, the probability order is computed first and the rank of each unit with $p_i > 0.9$ is reserved; these are ranks among units, independent of token cost. Each part's cost-weighted density $d(B')$ is computed from Equation (4), including its reserved units. Parts are sorted by decreasing density with ties broken by earliest original position, units within a part by decreasing p_i with the same positional tie rule; reserved units are removed from the concatenated list and the remaining units fill the unreserved ranks in order. This is done separately for Q_{in} and Q_{out}.

The saturation rule preserves a strong proposal preference under the declared update convention; the threshold $p_i > 0.9$ is a design constant with no calibration interpretation. Reserved units can interrupt the adjacency of a bundle's remaining units, and the anchor split separates its inside and outside parts, so the text prior encourages grouping subject to these constraints. Cost normalization makes density comparable across parts of different size; it involves no knapsack solution and no additive model of F.

The authoritative output is the permutation together with the units and their costs. An interface requiring scalar scores can assign unit π_r the ordinal value $(n - r + 1)/n$ and copy it to the unit's tokens; these values encode rank, and re-sorting raw proposal weights after decoding would discard the text-enhanced order. Both decoders take $O(n \log n)$ sorting time in a direct implementation, plus a linear text-segmentation pass for the text-enhanced one, and neither requires another target evaluation.

A.4 BM25 QUERY MATCHING WITH THE ANCHOR RELEASED

The query-matching control of Section 4.3 and Appendix A adds a different prior, lexical relevance to the public question. It uses the question as query and the source sentences containing eligible units as documents; sentence statistics are computed on the original public text, and only eligible units can enter the final intervention. The query excludes the generated target, reference answers and evidence annotations. Extraction prefers an explicitly marked question/problem field and otherwise a question-mark-terminated line; absent a valid public query, the query is empty. Case-folded lexical terms containing letters or digits are retained, and the fixed stop-word list is

```
a an the is are was were be been being and or of to in on at
for from by with as it its this that these those what which
who where when how do does did has have had can could would
should please answer question passage problem typical
special.
```

Numerals and within-word punctuation follow the lexical convention of the text processor, and query terms are deduplicated.

For N candidate sentences, let $f_{t,s}$ be the frequency of term t in sentence s, df_t its document frequency, ℓ_s the number of retained lexical terms in s, and $\bar{\ell} = \max(1, N^{-1}\sum_s \ell_s)$ for $N > 0$. The implementation uses the positive-IDF BM25 form (Robertson & Zaragoza, 2009)

$$v_s = \sum_{t \in \mathcal{Q}} \log\left(1 + \frac{N - \mathrm{df}_t + 0.5}{\mathrm{df}_t + 0.5}\right) \frac{f_{t,s}(k_1 + 1)}{f_{t,s} + k_1(1 - b + b\ell_s/\bar{\ell})}, \quad k_1 = 1.2, \quad b = 0.75. \tag{10}$$

An empty intervention domain returns the empty order. Otherwise, with $v_{\max} = \max_s v_s$, a unit assigned to sentence s receives $h_i = v_s / v_{\max}$ if $v_{\max} > 0$ and $h_i = 0$ if every sentence score is zero; an atom crossing a sentence boundary is assigned by its first source character. The adjusted weight is exactly Equation (5), $p'_i = p_i + 0.3 \max(\max_j p_j - \min_j p_j, 0.05) h_i$, an ordering score that may leave $[0, 1]$ and is left unclipped.

The control sorts all units globally by decreasing p'_i with positional ties, without the bundle/saturation decoder and without placing A_0 first. It can therefore admit units outside A_0 before units inside it and change $P_\pi(b_0)$; even when $h_i = 0$ everywhere, the fallback is global probability ordering, which likewise releases the anchor.

The control's hypothesis is that question-matching sentences tend to contain annotated evidence. This can improve localization while departing from the model's measured dependence, since the question's words need not identify what drove the fixed output. The comparison in Table 3 reports this divergence on a shared cohort. Because the control changes both the ordering signal and the anchor constraint, that comparison leaves the effect of BM25 reweighting alone unresolved; an anchor-preserving BM25 variant would be a separate diagnostic and would retain Recall@20% by construction when applied to the same search output.

Table 6: Three ways to decode the same search artifacts. All issue zero additional model queries. Only the first two preserve the searched intervention at b_0.

Ordering	Additional information	Anchor and intended role
Search only	Proposal weights and source positions	Preserved; extend the mask search to a complete order.
Text enhanced	Word/clause structure and offsets	Preserved; regularize the order within each anchor block.
BM25 control	Public question and sentence term statistics	Released; favor lexical relevance and probe localization versus faithfulness.

A.5 WHEN COULD HUMAN ALIGNMENT MAKE LINGUISTIC DECODING REDUNDANT?

A.5.1 THE HYPOTHESIS AND THE RELEVANT NOTION OF ALIGNMENT

A plausible hypothesis is that a model strongly aligned with human linguistic and task semantics already exposes coherent source dependencies through its response to intervention. Perturbing pieces

of the same meaningful expression would then affect the selected target in related ways, allowing the elite-mask updates to produce a stable order without an external grouping rule; the model's scoring behavior would supply the structure that the text-enhanced decoder otherwise imposes, and probability ordering could match it while avoiding a surface segmentation prior that may be inappropriate for the domain.

The relevant property is *intervention-level agreement with linguistic structure*, a stronger condition than fluent explanation or instruction following. Human preference alignment on ordinary text leaves open how a model responds when tokens are replaced by visible padding, and whether a population of 32 masks over five rounds reveals every useful within-set distinction. The proposed relationship between alignment and decoder redundancy is therefore an empirical hypothesis about models rather than a consequence of either definition.

The scope of redundancy also matters. At the searched budget, both anchor-preserving decoders return A_0 for every model, so linguistic decoding is already unnecessary there. The substantive question is whether linguistic structure improves the ordering inside and beyond A_0, and hence performance at other budgets. Here "mask optimization alone" means CEM followed by the minimal deterministic ordering of Equation (9), since CEM by itself returns a set and proposal weights.

A.5.2 A SUFFICIENT CONDITION FOR EXACT REDUNDANCY

The following condition, a property of the realized proposal weights and bundle partition, guarantees that the linguistic decoder changes nothing.

Proposition A.1 (Separated bundle ranges)**.** *Fix the same (U, c, A_0, p) for both anchor-preserving decoders. In each anchor block, suppose its nonempty bundle parts can be ordered $B_1, \dots, B_q$ so that*

$$\min_{u_i \in B_j} p_i > \max_{u_i \in B_{j+1}} p_i \qquad \text{for every } j < q. \tag{11}$$

If both decoders use the same within-part tie rule, then $\pi_{\text{text}} = \pi_{\text{prob}}$, including under the saturation rule.

Proof. Every density is a positive-cost weighted average and lies between the minimum and maximum proposal weights of its part, so Equation (11) orders the part densities in the same order as all their constituent weights. Concatenating the parts with decreasing weights within each part is exactly the probability order of the block, and removing the reserved units and reinserting them at their original probability ranks leaves this sequence unchanged. The argument applies to both anchor blocks, whose concatenation is shared. □

An illustrative regime is $p_i = \mu_j + \varepsilon_i$ for $u_i \in B_j$, with $|\varepsilon_i| \le \delta$ and $\mu_j - \mu_{j+1} > 2\delta$: within-part variation is then too small to mix the parts' ranges, and the proposition applies. This is a diagnostic model for the realized weights, independent of any statistical assumption of CEM, and shows how coherent, well-separated proposal weights make grouping redundant. Exact redundancy also occurs when every unit is reserved by saturation. Conversely, identical rankings can arise from uninformative ties, so decoder agreement by itself is uninformative about alignment or faithfulness.

Removing the anchor partition requires an additional condition. When both anchor blocks are nonempty, $\min_{i \in A_0} p_i > \max_{i \notin A_0} p_i$ ensures that a global probability sort also places the entire anchor first; without it, dropping the linguistic prior gives no reason to discard the verified set A_0.

A.5.3 WHY ALIGNMENT NEED NOT ELIMINATE THE PRIOR

Finite search can leave the internal order underdetermined even for a simple, consistent score. Take ten unit-cost units, $b_0 = 2$, and

$$F(A) = \mathbf{1}[u_1 \in A] + 9\mathbf{1}[u_2 \in A]. \tag{12}$$

If every round's elite multiset contains only $\{u_1, u_2\}$, the inclusion frequencies and logits of the two units remain equal from the common initialization, and position-based ties place u_1 before u_2 although their singleton effects are 1 and 9. The search has identified the best two-unit set without identifying the best first unit. This is a possible finite-search outcome under the stated sampling rule; it shows that even an additive objective need not make proposal weights recover contribution

magnitudes. Textual grouping does not resolve this example either, since its benefit depends on relevant structural information.

More generally, redundant expressions can substitute for one another in elite masks, complementary units can matter only jointly, and a negation or numeral can make one token within an otherwise coherent phrase decisive. Alignment with human semantics is compatible with all of these effects. The text decoder may reduce fragmentation when grouping is appropriate or introduce bias when a surface bundle combines units with different roles; the saturation rule limits some rearrangements, and whether the remaining ones improve faithfulness is an empirical matter. A larger query budget reduces sampling variability, yet a single-budget objective identifies other budgets only under further assumptions.

The hypothesis reads differently for BM25. Linguistic grouping supplies a prior over *local source structure*, whereas BM25 supplies a prior over *relevance to the question* and releases the anchor. Alignment that makes the former redundant leaves the latter open, as it leaves open whether reference-evidence localization coincides with deletion faithfulness; conversely, higher recall under BM25 shows only how a question-matching order fares under the annotation metric.

A.5.4 A FALSIFIABLE EVALUATION OF THE HYPOTHESIS

The hypothesis should be evaluated on model families or checkpoints whose alignment is measured independently of these attribution outcomes. Matched checkpoints with the same architecture and tokenizer are preferable, scoring the same fixed transcript, source units and target at each checkpoint; an evaluation on model-generated output changes the target and is reported separately. Sequence length, source–target separation, replacement protocol and search budget are controlled, and within each checkpoint and seed the same (A_0, p) is decoded both ways, so that a decoder effect is never confounded with a new search.

The primary comparison is the paired difference in deletion AUC and in off-anchor budget effects between π_{text} and π_{prob}, with a pre-specified equivalence margin and paired uncertainty estimates over source problems, since an undetected difference is distinct from equivalence. Seed variation, proposal ties, the fraction of reserved units, within-bundle weight variation and decoder ranking disagreement distinguish coherent weights from degenerate agreement, and varying the search budget as a separate factor separates model alignment from the amount of search evidence. An aligned model can make the prior redundant by matching or exceeding its faithfulness without satisfying the sufficient condition above, which serves as a transparent diagnostic rather than a necessary criterion.

BM25 requires a further separation of effects: probability and BM25-adjusted weights compared both with and without the anchor constraint, holding the search fixed, which isolates lexical reweighting from anchor release. This diagnostic is proposed here, with localization and faithfulness evaluated separately on matched cohorts and annotations used only for evaluation. The decoder and query-matching experiments of Appendix C.8 and Section 4.3 measure decoder and query-matching performance; alignment enters only in the evaluation proposed above. Support for the hypothesis would require strong alignment together with a reproducibly negligible benefit from the linguistic decoder; a persistent positive decoder benefit under the controlled comparison would bound it.

B IMPLEMENTATION DETAILS AND EXPERIMENTAL SETTINGS

B.1 EXPERIMENTAL SETTINGS

All experiments use a single NVIDIA RTX PRO 6000 Blackwell Max-Q GPU, with PyTorch `2.11.0+cu128` and Transformers `5.8.1`. Qwen3-8B and Qwen3.5-9B use their native chat templates with thinking disabled and greedy generation. Generation and deletion evaluation use BF16; Integrated Gradients uses FP32. Generation and likelihood scoring use native SDPA where supported, whereas information-flow methods and DePASS use native eager attention. Qwen3.5's recurrent layers use the PyTorch delta-rule implementation.

The generated token sequence and attribution target are fixed before evaluation and shared across methods and deletion budgets. Table 7 gives the generation limits and targets; eligibility criteria and metric cohorts are specified in Appendix D. The supplied mathematical prefix remains part of the input.

Table 7: Maximum generation lengths and fixed attribution targets.

Task	Maximum new tokens	Target
NIAH	64	Generated answer
Multi-hop QA	96	Generated answer
Mathematics	1024	Full visible generated continuation
MCQ	32	Generated option-letter span
Code	2048	Full generated code target

All methods are evaluated on the same public candidate region and indivisible units, with padding replacement and strict-prefix deletion as defined in Appendices A.1 and E. Scalar token scores are averaged within each unit and ranked decreasingly, retaining their signs and breaking ties by source order. TEMPO restricts its search to the candidate units; information-flow methods propagate through the full computation and filter their output scores to those units. Reference-evidence annotations are used only for evaluation.

B.2 Baseline implementations

For gradient and leave-one-out methods, $\mathcal{L}$ denotes the sum of fixed-target log probabilities; reported deletion effects use the per-token mean.

IFR and FlashTrace. IFR (Ferrando & Voita, 2024) uses the tokenwise path of the FlashTrace implementation, averaging importance over target positions. FlashTrace (Pan et al., 2026b) uses span aggregation with one recursive hop. Both use the actual generated IDs, native attention, chunk size 64, sink chunk size 8 and renormalization threshold zero. Their nonnegative scores measure information flow to generated-token representations.

Gradient $\times$ Input and IG. Gradient $\times$ Input (Ancona et al., 2018) returns $\langle E_i, \nabla_{E_i}\mathcal{L}\rangle$. IG (Sundararajan et al., 2017) follows a straight path from padding embeddings at user positions, keeping system/template positions and generated history fixed. Adaptive Simpson integration uses FP32, an initial setting of 32 nodes, a maximum of 512 nodes, and absolute and relative tolerances of 0.05. Budget-limited runs are retained as unconverged, without rescaling scores to enforce completeness.

DePASS. We adapt the full-token softmax decomposition (Hong et al., 2025) to native model operations, using temperature 0.1 and one component per sequence token plus a remainder. The readout sums target-token logit contributions at their prediction positions. For Qwen3.5, the available FlashTrace path fails to reconstruct the full hybrid mixer, and DePASS lacks a hybrid-layer decomposition; these baselines are therefore evaluated only on Qwen3-8B.

AttriBoT. We evaluate the lossless prefix-cache leave-one-out component (Liu et al., 2025), adapted to token padding replacement with batch size eight. Scores are decreases in $\mathcal{L}$. The unchanged prefix is reused and the intervened suffix recomputed, including recurrent states on Qwen3.5. Hierarchical, proxy-model and pruning accelerations are not used.

CAMAB. Linear Thompson sampling (Pan et al., 2026a) is adapted to token retention, with unselected user tokens padded. It uses 40 iterations, an intercept, a full Gaussian posterior, prior mean zero, prior variance one and observation-noise variance 0.01. The reward is mean target log probability; scores are posterior-mean token coefficients excluding the intercept.

B.3 TEMPO configuration

Search and decoding use the parameters in Appendices A.2 and A.3.2. The executed update substitutes b_0/C for the nominal ρ_0 in Equation (3); summed target log probabilities are used during search, preserving candidate order for the fixed target. TEMPO and CAMAB use random seed 20260917. Search candidates are scored individually, and dense deletion evaluation uses batches of four masks. The reported decoder applies strict $p_i > 0.9$ rank protection. Search-only and BM25 controls reuse the same search output as specified in Appendices A.3.1 and A.4.

B.4 Timing and statistical reporting

Attribution time excludes generation and deletion evaluation, charging TEMPO's full search cost even when search artifacts are reused. Qwen3-8B main and code results retain historical single-run attribution timings, while the current decoder was re-evaluated for faithfulness. Qwen3.5 TEMPO times combine recorded same-input search time with the median of three CPU decoder timings after one warm-up. Confidence intervals follow the paired bootstrap procedure in Appendix D, with RNG seed 20260922.

C Supplementary Experiments

C.1 Per-Task Results on Qwen3-8B

This section gives the full version of the comparison that Table 2 shows for three methods: per-task values for the four tasks are in Table 8, the signed drop at the three fixed budgets behind Table 1 is in Figure 5, and both are discussed in Section 4.2 and Section 4.3. Beyond the per-task AUC and D@20 of Table 1 with the four-task macro, this section adds all eight methods across all four task families, the remaining fixed budgets, the localization columns and the budget sweep of Figure 5.

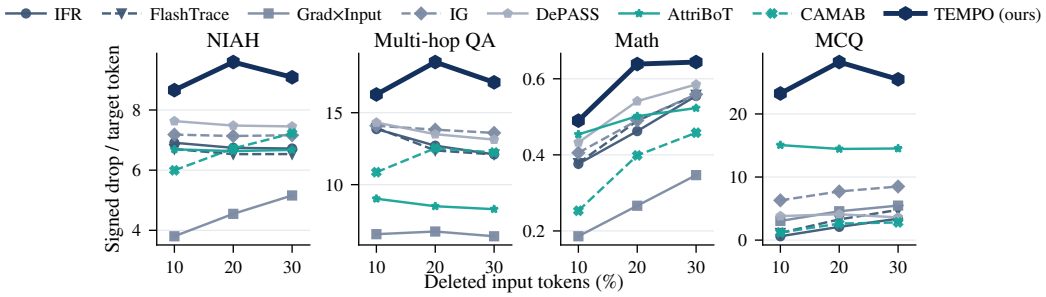

Figure 5: Signed drop per target token on Qwen3-8B at the 10%, 20% and 30% budgets, each a fraction of the total token cost of the intervention units, averaged over each task's inputs (the per-task values behind Table 1; Appendix C.1). 20% is the budget at which TEMPO's search operates, and TEMPO is above all seven baselines at all three budgets on every task.

Table 8: Per-task results on Qwen3-8B (full cohort). Table 2 in the main text gives the corresponding localization columns for three of these methods; this table gives all eight. NIAH pools the short- and long-needle variants. Faithfulness n: NIAH 179, multi-hop 51, math 79, MCQ 94; localization n: NIAH 200, multi-hop 100, math 39, MCQ 94 (433 in total). The time column is measured on the 473 inputs for which every method produced an attribution, not on the faithfulness cohort. Best per column in bold, and a column whose best value is attained by more than one method is left unbolded.

Task	Method	AUC	Drop@10%	Drop@20%	Drop@30%	Recall@20%	NDCG@10	Time (s)
NIAH	IFR (Ferrando & Voita, 2024)	6.36	6.91	6.74	6.72	**0.891**	0.999	2.84
	FlashTrace (Pan et al., 2026b)	6.18	6.70	6.53	6.53	0.849	0.999	0.67
	Grad×Input (Ancona et al., 2018)	5.36	3.80	4.55	5.16	0.332	0.312	**0.33**
	IG (Sundararajan et al., 2017)	6.98	7.18	7.13	7.16	0.763	0.971	198.7
	DePASS (Hong et al., 2025)	7.06	7.63	7.48	7.45	0.614	0.961	244.4
	AttriBoT (Liu et al., 2025)	6.65	6.69	6.63	6.67	0.543	0.998	162.2
	CAMAB (Pan et al., 2026a)	6.59	5.99	6.72	7.22	0.309	0.369	6.06
	TEMPO	**8.00**	**8.66**	**9.59**	**9.09**	0.273	0.469	24.4
Multi-hop	IFR (Ferrando & Voita, 2024)	11.79	13.86	12.71	12.10	**0.520**	0.534	0.23
	FlashTrace (Pan et al., 2026b)	11.91	13.95	12.38	12.10	0.488	0.533	0.18
	Grad×Input (Ancona et al., 2018)	7.07	6.56	6.74	6.41	0.331	0.400	**0.11**
	IG (Sundararajan et al., 2017)	12.63	14.09	13.82	13.60	0.392	0.407	38.0
	DePASS (Hong et al., 2025)	12.37	14.32	13.51	13.13	0.357	0.482	9.45
	AttriBoT (Liu et al., 2025)	9.08	9.02	8.50	8.30	0.452	**0.593**	6.10
	CAMAB (Pan et al., 2026a)	11.57	10.86	12.53	12.24	0.255	0.289	2.16
	TEMPO	**14.04**	**16.27**	**18.52**	**17.11**	0.249	0.342	8.68
Math	IFR (Ferrando & Voita, 2024)	0.592	0.376	0.462	0.555	0.308	0.187	9.87
	FlashTrace (Pan et al., 2026b)	0.597	0.378	0.490	0.558	0.326	0.175	0.29
	Grad×Input (Ancona et al., 2018)	0.442	0.186	0.266	0.347	0.193	0.090	**0.12**
	IG (Sundararajan et al., 2017)	0.592	0.405	0.491	0.559	0.362	0.224	38.5
	DePASS (Hong et al., 2025)	0.605	0.433	0.541	0.585	0.368	0.182	16.6
	AttriBoT (Liu et al., 2025)	0.571	0.454	0.502	0.523	**0.375**	**0.256**	5.22
	CAMAB (Pan et al., 2026a)	0.538	0.253	0.399	0.458	0.197	0.129	2.47
	TEMPO	**0.629**	**0.490**	**0.639**	**0.644**	0.254	0.189	9.91
MCQ	IFR (Ferrando & Voita, 2024)	3.58	0.61	2.10	3.40	0.006	0.000	0.15
	FlashTrace (Pan et al., 2026b)	3.77	1.13	3.28	4.79	0.027	0.000	0.35
	Grad×Input (Ancona et al., 2018)	4.77	3.04	4.57	5.47	0.266	0.038	**0.10**
	IG (Sundararajan et al., 2017)	6.34	6.30	7.71	8.50	0.355	0.089	57.6
	DePASS (Hong et al., 2025)	3.65	3.81	4.10	3.60	0.208	0.032	5.84
	AttriBoT (Liu et al., 2025)	11.08	15.06	14.45	14.52	**0.722**	**0.672**	3.85
	CAMAB (Pan et al., 2026a)	3.87	1.20	2.62	2.80	0.077	0.000	2.15
	TEMPO	**17.33**	**23.25**	**28.23**	**25.49**	0.656	0.656	8.56

C.2 NIAH BY NEEDLE LENGTH

Table 9 adds the per-variant values and the cohort of each metric behind the short- and long-needle comparison of Section 4.3. NIAH pairs every source problem with a short (11–23-token) and a long (53–65-token) needle, so reference-evidence size varies while question and context stay fixed. Recall@20% and SBα-NDCG@10 are computed on the 100 short and 100 long inputs with annotations, while the AUC and Drop@20% columns use the 95 short and 84 long inputs with a valid fixed target; the table reports all four metrics, with Recall@20% primary. Faithfulness barely moves. Every method's AUC rises by 0.1–0.34 from short to long needles, the Recall@20% ordering is unchanged, and TEMPO leads both variants on AUC and Drop@20%. SBα-NDCG@10, saturated once the first needle tokens are found, stays at 0.96–1.00 for IFR, FlashTrace, IG, DePASS and AttriBoT, whereas Recall@20% falls for every method. A long needle is located as reliably as a short one yet covered less within a 20% budget, because token recall rewards the whole evidence segment while deletion faithfulness rewards the tokens the answer depends on.

Table 9: NIAH by needle length on Qwen3-8B; best per column in bold, and a column whose best value is attained by more than one method is left unbolded.

	Recall@20%↑		**NDCG@10↑**		**AUC↑**		**Drop@20%↑**	
Method	short	long	short	long	short	long	short	long
IFR (Ferrando & Voita, 2024)	**0.984**	**0.798**	**0.999**	0.999	6.30	6.42	6.65	6.85
FlashTrace (Pan et al., 2026b)	0.935	0.762	0.998	0.999	6.13	6.24	6.47	6.61
Grad×Input (Ancona et al., 2018)	0.361	0.304	0.327	0.297	5.20	5.54	4.45	4.66
IG (Sundararajan et al., 2017)	0.878	0.648	0.971	0.971	6.91	7.05	7.06	7.22
DePASS (Hong et al., 2025)	0.709	0.520	0.961	0.961	6.97	7.17	7.37	7.61
AttriBoT (Liu et al., 2025)	0.657	0.430	0.998	0.999	6.50	6.81	6.55	6.73
CAMAB (Pan et al., 2026a)	0.361	0.256	0.341	0.397	6.51	6.69	6.68	6.77
TEMPO	0.307	0.240	0.457	0.481	**7.88**	**8.14**	**9.41**	**9.80**

C.3 MATH: MISLEADING AND CORRECT PREFIXES

This section adds the split of the math family by outcome and the per-side values behind the narrow lead on math reported in Section 4.2. The 40 source problems of the math family separate by outcome: the injected prefix changes the generated answer in 23, both variants answer correctly in 13, both answer incorrectly in 2, the prefix changes the answer the other way in one, and one generated response is truncated and cannot be judged. Table 10 reports the 23 pairs in which the misleading prefix produces a wrong answer while the same problem with a correct prefix is answered correctly. The two sides generate different answers, so each is scored against its own fixed target, giving a paired comparison of the two sides. Recall@20% and SBα-NDCG@10 exist only on the misleading side, which carries the error-source annotation. TEMPO leads every faithfulness column on both sides, by 0.034 AUC against DePASS on the misleading side and by 0.036 on the correct side, and AttriBoT leads recall and NDCG on the misleading side (Appendix C.4 reports the corresponding split for the multiple-choice family).

Table 10: Math on Qwen3-8B over the 23 source problems whose injected misleading prefix changes the generated answer; each side is scored against its own generated target. Recall@20% and SBα-NDCG@10 are reported only for the misleading side, which carries the error-source annotation. Best per column in bold.

	Misleading prefix (wrong answer)						Correct prefix (correct answer)			
Method	AUC↑	D@10%↑	D@20%↑	D@30%↑	R@20%↑	NDCG@10↑	AUC↑	D@10%↑	D@20%↑	D@30%↑
IFR	0.859	0.578	0.703	0.828	0.255	0.106	0.877	0.590	0.703	0.848
FlashTrace	0.866	0.587	0.731	0.840	0.267	0.090	0.888	0.590	0.768	0.852
Grad×Input	0.641	0.338	0.416	0.504	0.175	0.063	0.631	0.230	0.379	0.506
IG	0.853	0.588	0.718	0.817	0.428	0.241	0.893	0.654	0.760	0.862
DePASS	0.871	0.646	0.807	0.861	0.428	0.157	0.902	0.694	0.843	0.908
AttriBoT	0.829	0.667	0.741	0.783	**0.443**	**0.262**	0.842	0.702	0.763	0.776
CAMAB	0.765	0.371	0.621	0.667	0.194	0.107	0.803	0.413	0.584	0.690
TEMPO	**0.905**	**0.714**	**0.938**	**0.940**	0.322	0.200	**0.938**	**0.755**	**0.951**	**0.958**

C.4 MCQ: Original and Counterfactual Variants

This section adds the original/counterfactual split behind the MCQ columns of Table 2 and the MCQ claim of Section 4.3. Each MCQ source problem has an original and a counterfactual variant in which one value token is edited so that the correct option changes; both are generated, attributed and evaluated independently under padding deletion. Table 11 reports the two variants separately on all valid inputs (46 original, 48 counterfactual) and on the 45 source problems for which both variants are valid. The two variants agree on the leaders: TEMPO has the highest AUC on both (17.66 / 17.02) and AttriBoT the highest Recall@20% on both (0.72 / 0.73), while the middle of the ranking moves, since DePASS and Grad×Input swap places on recall (0.244 against 0.242 on the original variant, 0.173 against 0.289 on the counterfactual). The information-flow methods recover almost none of the decisive value ($\leq$ 0.04 recall) on either variant. On NDCG@10 the two variants differ: AttriBoT leads on the counterfactual variant (0.698 against TEMPO's 0.649), while on the original variant TEMPO leads (0.665 against 0.646). The split therefore leaves the leader of each axis unchanged and moves only the middle of the ranking, which is the same divergence between the two axes that Section 4.3 reports.

Table 11: MCQ on Qwen3-8B by variant. Each AUC cell reports two values, all valid inputs / the 45 source problems for which both variants are valid; Recall@20% and SBα-NDCG@10 are on all valid inputs (46 original, 48 counterfactual). Best per column in bold, counting both values of an AUC cell, and a column whose best value is attained by more than one method is left unbolded.

	Original (46; paired 45)			Counterfactual (48; paired 45)		
Method	AUC	R@20%	NDCG	AUC	R@20%	NDCG
IFR (Ferrando & Voita, 2024)	3.89 / 3.87	0.000	0.000	3.29 / 3.41	0.011	0.000
FlashTrace (Pan et al., 2026b)	3.97 / 3.95	0.039	0.000	3.58 / 3.71	0.016	0.000
Grad×Input (Ancona et al., 2018)	4.98 / 4.93	0.242	0.046	4.57 / 4.68	0.289	0.030
IG (Sundararajan et al., 2017)	6.15 / 6.12	0.320	0.104	6.52 / 6.84	0.388	0.075
DePASS (Hong et al., 2025)	3.48 / 3.48	0.244	0.038	3.82 / 3.85	0.173	0.026
AttriBoT (Liu et al., 2025)	11.12 / 11.16	**0.719**	0.646	11.05 / 11.47	**0.725**	**0.698**
CAMAB (Pan et al., 2026a)	4.15 / 4.06	0.098	0.000	3.59 / 3.73	0.057	0.000
TEMPO	**17.66 / 17.81**	0.669	**0.665**	**17.02 / 17.42**	0.644	0.649

C.5 Held-Out Source Problems

This section supplies the construction of the extension set, its per-task values and the paired differences behind the extension-set claim of Section 4.2. The extension set removes every input whose source

problem was used to select the decoder, leaving 400 inputs from 240 sources, of which 335 have a valid faithfulness target and 305 also have annotations. No hyper-parameter was changed, so the frozen configuration is evaluated on source problems that decoder selection did not use. Table 12 reports the four-task macro on this set (Section 4.2).

Table 12: Extension set (Qwen3-8B, decoder-selection sources removed), four-task macro. Faithfulness on 335 inputs; localization on 305. Best per column in bold, and a column whose best value is attained by more than one method is left unbolded.

Method	AUC	Drop@10%	Drop@20%	Drop@30%	Recall@20%	NDCG@10	Time (s)
IFR	5.45	5.36	5.35	5.44	0.431	0.433	3.31
FlashTrace	5.46	5.36	5.40	5.79	0.421	0.429	0.39
Grad×Input	4.33	3.28	3.98	4.35	0.264	0.206	**0.17**
IG	6.53	6.97	7.11	7.41	0.461	0.418	84.6
DePASS	5.74	6.29	6.20	6.04	0.393	0.419	69.3
AttriBoT	6.84	7.97	7.61	7.46	**0.530**	**0.636**	44.5
CAMAB	5.54	4.52	5.47	5.68	0.209	0.197	3.28
TEMPO	**9.92**	**12.18**	**14.24**	**13.02**	0.356	0.420	13.1

Per task, TEMPO's AUC on this set is 7.93 (NIAH, 161 inputs; best baseline DePASS 7.02), 13.51 (multi-hop, 41; IG 12.07), 0.508 (math, 59; DePASS 0.478) and 17.72 (MCQ, 74; AttriBoT 11.30). The paired macro AUC differences are $+3.08$ $[2.62, 3.56]$ against AttriBoT, $+0.97$ $[0.75, 1.19]$ against the search-only ordering and $+2.08$ $[1.79, 2.37]$ against the query-matching control.

C.6 Search-Seed Stability

Three additional search seeds (20260924, 20260925, 20260926) were run alongside the original seed (20260917) on the 80 inputs used to select the decoder (68 valid faithfulness inputs, 70 localization inputs per seed), with the generated targets, search hyper-parameters, intervention and decoder held fixed; only the search's random draws change (Section 4.2). Table 13 reports every seed with its per-task dispersion, the per-seed values behind the stability claim of Section 4.2. The four-seed macro AUC is 10.17 ± 0.28 (sample SD; range 9.80–10.44); the largest task-level dispersion is on MCQ (SD 1.06), the smallest on math (SD 0.011). The paired macro AUC difference against the search-only ordering is positive for every seed: $+1.15$, $+0.64$, $+1.06$ and $+0.76$. The search is stable across seeds for a fixed generation and a fixed input set; a seed SD is not a bootstrap interval.

Table 13: Search-seed stability on the 80 inputs used to select the decoder (Qwen3-8B). Macro over four tasks; the search-only column gives the same seed's search-only ordering. ΔAUC is printed as computed from the underlying values, so at the printed precision it need not equal the difference of the two AUC columns. The 10.44 of seed 20260917 is this stability run. Best value per column among the four seed rows in bold (higher is better in every column); the Mean ± SD row is a summary of the seeds and is not bolded.

Seed	AUC	D@10%	D@20%	D@30%	R@20%	NDCG@10	Search-only AUC	ΔAUC
20260917	**10.44**	**12.25**	14.35	**13.48**	**0.362**	0.398	9.29	**+1.15**
20260924	9.80	12.10	14.12	12.58	0.335	**0.410**	9.15	+0.64
20260925	10.33	11.89	**14.41**	13.45	0.339	0.403	9.27	+1.06
20260926	10.13	11.31	14.40	13.41	0.354	0.376	**9.37**	+0.76
Mean ± SD	10.17 ± 0.28							

Per task, the four-seed AUC means and SDs are NIAH 8.43 ± 0.17, multi-hop 16.09 ± 0.29, math 0.987 ± 0.011 and MCQ 15.18 ± 1.06.

C.7 Efficiency and Short-Input Timing

This section adds the per-task times and the access requirement of every method behind the time claim of Section 4.2.

Table 14 reports attribution time against faithfulness for every method. TEMPO takes 12.9 s per input on average, of which decoding is about 9 ms. This is $3.4\times$ less than the AttriBoT implementation (44.3 s), $5.4\times$ less than DePASS (69.1 s) and $6.5\times$ less than IG (83.2 s); on NIAH the gap widens

Table 14: Attribution time per input in seconds on Qwen3-8B, by task and as the four-task macro (per-task rows of Table 8), with macro AUC from Table 1. Macro times are computed from the unrounded per-task values, so at printed precision they need not equal their mean. Times are single-run wall-clock measurements on shared hardware, attribution only. Bold marks the fastest macro time and the highest macro AUC; the fastest time in each per-task column is bolded as well.

Method	Access	NIAH	Multi-hop	Math	MCQ	Macro	Macro AUC↑
IFR	internals	2.84	0.23	9.87	0.15	3.27	5.58
FlashTrace	internals	0.67	0.18	0.29	0.35	0.37	5.61
Grad×Input	gradients	**0.33**	**0.11**	**0.12**	**0.10**	**0.17**	4.41
IG	gradients	198.7	38.0	38.5	57.6	83.2	6.64
DePASS	internals	244.4	9.45	16.6	5.84	69.1	5.92
AttriBoT	scoring	162.2	6.10	5.22	3.85	44.3	6.84
CAMAB	scoring	6.06	2.16	2.47	2.15	3.21	5.64
TEMPO	scoring	24.4	8.68	9.91	8.56	12.9	**10.00**

to 162–244 s for those three. FlashTrace, Grad×Input, IFR and CAMAB are faster; TEMPO is the cheapest model-agnostic method that is at least as faithful as the strongest baseline, and its cost is a fixed number of scoring calls, independent of the number of sources. A second timing workload on Llama-3.1-8B, twenty inputs whose user text is truncated to 256 tokens, with 16-token targets and one CUDA-synchronised measurement per method and input (Appendix C.7.1), gives the short-input end of the picture: TEMPO takes 5.75 s against 74.6 s for IG, while AttriBoT (2.70 s) and DePASS (4.20 s) are cheaper when the number of sources is this small.

C.7.1 Timing on Llama-3.1-8B

This section adds the per-task and macro times behind the time claim of Section 4.2, measured on a second model family and on short inputs. To place every method, including IG, on one timing workload outside the Qwen family, we time the first five inputs of each task in the 80-input plan (20 inputs) on the base meta-llama/Llama-3.1-8B checkpoint with a completion template, on an NVIDIA RTX PRO 6000 Blackwell Max-Q GPU. User text is truncated to at most 256 tokens and re-encoded, generation is capped at 16 tokens, all methods share the same frozen generated prefix, and each pair of method and input has one warm-up scoring call and one CUDA-synchronised timed measurement (loading, generation and warm-up excluded; original method budgets; DePASS with 4×4 tiling). Answer filtering and the option-only MCQ target are not applied to this timing workload, and the truncation removes task evidence, so the workload measures computational cost on short inputs only. Table 15 reports the per-task and macro times and their spread across inputs.

Table 15: Attribution time in seconds on Llama-3.1-8B, 20 short inputs (five per task; prompts of 133–272 tokens, targets of 3–16 tokens). SD is across inputs, not across repetitions. Bold marks the fastest macro time and the fastest time in each per-task column.

Method	NIAH	Multi-hop	Math	MCQ	Macro	SD (20 inputs)
IFR (Ferrando & Voita, 2024)	0.40	0.35	0.21	0.30	0.32	0.10
FlashTrace (Pan et al., 2026b)	0.15	0.14	0.11	0.13	0.13	0.02
Grad×Input (Ancona et al., 2018)	**0.09**	**0.07**	**0.06**	**0.07**	**0.07**	0.02
IG (Sundararajan et al., 2017)	104.14	82.08	59.85	52.16	74.56	31.54
DePASS (Hong et al., 2025)	5.12	5.10	2.17	4.43	4.20	1.29
AttriBoT (Liu et al., 2025)	3.38	3.34	1.27	2.82	2.70	0.91
CAMAB (Pan et al., 2026a)	1.57	1.58	1.03	1.40	1.39	0.26
TEMPO	6.47	6.52	4.26	5.75	5.75	1.05

IG completed all 20 inputs; 18 met its joint convergence criterion and two NIAH inputs reached the 512-node budget (509 nodes evaluated) without converging, and their times are included. The five NIAH records yield four distinct encoded prompts after truncation. On this short workload TEMPO costs 5.75 s against 74.6 s for IG: with $\leq$ 272 prompt tokens, leave-one-out needs at most a few hundred forward passes, fewer than TEMPO's fixed budget of 160 candidate scores, whereas on the full-length NIAH inputs of Appendix C.7 the same methods cost 162–244 s against 24 s.

C.8 Decoder Ablation

This section adds the per-task breakdown of the two TEMPO variants and their paired differences behind the decoder-ordering comparison of Section 4.4 and Figure 4. To isolate the contribution of text-enhanced decoding we order the *same* search output without text structure: the searched set A_0 first, then all remaining units, each block sorted by the proposal weight p_i ("search only"; this is the anchor-first ordering of the underlying mask-search method (Gu et al., 2026) adapted to text units). Because both orderings share A_0, Drop@20% and Recall@20% are identical on every input (Section 3.2); the comparison therefore measures ordering quality inside and beyond the searched set.

Table 16: Search-only ordering, TEMPO and the query-matching control on Qwen3-8B (same search output; 403 faithfulness inputs, 433 localization inputs). Macro and per-task values; Drop@20% and Recall@20% coincide for the two TEMPO variants by construction. Bold marks the higher of the two TEMPO variants and a column whose two values are equal is left unbolded; the control is listed for the three-way comparison. Its row uses the same metric-specific cohorts as the ablation; Table 3 compares it with TEMPO on the joint cohort.

	Macro						
Ordering	AUC	D@10%	D@20%	D@30%	R@20%	NDCG@10	Time (s)
Search only	9.00	11.96	14.24	12.86	0.358	0.414	12.9
TEMPO	**10.00**	**12.17**	14.24	**13.08**	0.358	0.414	12.9
Query-matching control	7.88	12.07	12.80	10.99	0.432	0.560	12.9
	NIAH						
Search only	7.66	8.41	9.59	9.01	0.273	0.469	24.4
TEMPO	**8.00**	**8.66**	9.59	**9.09**	0.273	0.469	24.4
Query-matching control	7.10	8.80	8.65	7.23	0.440	0.868	24.4
	Multi-hop						
Search only	13.54	15.92	18.52	16.97	0.249	0.342	8.7
TEMPO	**14.04**	**16.27**	18.52	**17.11**	0.249	0.342	8.7
Query-matching control	12.33	15.80	16.68	15.26	0.345	0.483	8.7
	Math						
Search only	0.620	0.480	0.639	**0.647**	0.254	0.189	9.9
TEMPO	**0.629**	**0.490**	0.639	0.644	0.254	**0.189**	9.9
Query-matching control	0.597	0.468	0.583	0.602	0.277	0.210	9.9
	MCQ						
Search only	14.17	23.04	28.23	24.81	0.656	0.656	8.6
TEMPO	**17.33**	**23.25**	28.23	**25.49**	0.656	**0.656**	8.6
Query-matching control	11.50	23.22	25.31	20.85	0.667	0.680	8.6

Paired differences. The macro AUC difference (TEMPO minus search only) is $+1.00$ ($+11.2\%$) with a source-clustered bootstrap 95% interval of $[0.81, 1.20]$; Drop@10% changes by $+0.20$ $[-0.03, 0.45]$ and Drop@30% by $+0.22$ $[0.03, 0.44]$; Drop@20% and Recall@20% are identical on every input; and SBα-NDCG@10 differs by $+0.0003$ $[-0.002, 0.002]$ on the 433 localization inputs: fixing saturated units at their proposal-order ranks preserves the top-rank agreement of the proposal-weight order while the bundle ordering improves the rest of the curve. On the 363-input joint cohort the AUC difference is $+1.00$ $[0.81, 1.21]$, and on the extension set $+0.97$ $[0.75, 1.19]$. Without the saturation rule the decoder gave an AUC difference of $+0.89$ $[0.70, 1.08]$ and lowered NDCG@10 by 0.136; the rule recovers the top-rank agreement and adds $+0.11$ $[0.08, 0.15]$ AUC

over that revision. The saturation threshold $p_i > 0.9$ is fixed across every reported result. Across four search seeds the difference is $+1.15$, $+0.64$, $+1.06$ and $+0.76$ (Appendix C.6).

Query-matching control vs. search only. Relative to the search-only ordering, the control changes AUC by -1.11 $[-1.27, -0.95]$ (-12.4%), Drop@20% by -1.44 $[-1.64, -1.24]$, Recall@20% by $+0.074$ $[+0.064, +0.086]$ ($+20.7\%$) and SBα-NDCG@10 by $+0.146$ $[+0.127, +0.165]$ ($+35.2\%$). These differences are computed on the metric-specific cohorts of Section 4.1: the AUC and Drop@20% differences on the 403-input faithfulness cohort, the Recall@20% and SBα-NDCG@10 differences on the 433-input localization cohort. Table 3 reports this control against TEMPO rather than against the search-only ordering, and the cohort of each row is given where the control is discussed (Section 4.3).

Table 17: Plotted coordinates of Figure 3 on Qwen3-8B: signed-drop AUC and Recall@20% per task, with the query-matching control. Best per column among the eight methods in bold; the control is shown because the arrow of Figure 3 is drawn to it and is not ranked. On NIAH and multi-hop the TEMPO recall values are joint-cohort, so they differ from Table 8.

	NIAH		Multi-hop		Math		MCQ	
Method	AUC↑	R@20%↑	AUC↑	R@20%↑	AUC↑	R@20%↑	AUC↑	R@20%↑
IFR (Ferrando & Voita, 2024)	6.36	**0.891**	11.79	**0.520**	0.592	0.308	3.58	0.006
FlashTrace (Pan et al., 2026b)	6.18	0.849	11.91	0.488	0.597	0.326	3.77	0.027
Grad×Input (Ancona et al., 2018)	5.36	0.332	7.07	0.331	0.442	0.193	4.77	0.266
IG (Sundararajan et al., 2017)	6.98	0.763	12.63	0.392	0.592	0.362	6.34	0.355
DePASS (Hong et al., 2025)	7.06	0.614	12.37	0.357	0.605	0.368	3.65	0.208
AttriBoT (Liu et al., 2025)	6.65	0.543	9.08	0.452	0.571	**0.375**	11.08	**0.722**
CAMAB (Pan et al., 2026a)	6.59	0.309	11.57	0.255	0.538	0.197	3.87	0.077
TEMPO	**8.00**	0.277	**14.04**	0.250	**0.629**	0.254	**17.33**	0.656
Query-matching control	7.10	0.440	12.33	0.345	0.597	0.277	11.50	0.667

C.9 Code Generation

This section adds the pool construction and the interval detail behind the code-generation paragraph of Section 4.4 and the table Table 5. For code we use the evaluation protocol of FlashTrace (Pan et al., 2026b) on our own pool: one greedy non-thinking generation per exercise capped at 2048 tokens, keeping only the programs that pass all their reference tests, with the entire generated program as the target. The pool is the Exercism Python track at revision `1f6aab8`: of the 140 exercises sampled with a fixed seed, 48 of the 139 evaluable generations passed all their reference tests, and one exercise is excluded because its reference test is skipped. Code attribution has no reference evidence, so only faithfulness is reported (Table 5).

The AUC interval against FlashTrace is $+0.028$ $[-0.054, 0.110]$ and includes zero, that against AttriBoT is $+0.197$ $[0.127, 0.274]$ and does not, and the search-only ordering is within 0.012 $[-0.011, 0.036]$ of TEMPO on this set.

D Experimental Setup Details

This appendix gives the baseline configurations, the data construction and cohorts, the annotation conventions and the protocol details behind the setup of Section 4.1.

Baselines. The baselines span four families: IFR (Ferrando & Voita, 2024) and FlashTrace (Pan et al., 2026b) (information flow), Grad×Input (Ancona et al., 2018) and Integrated Gradients

(IG) (Sundararajan et al., 2017), DePASS (Hong et al., 2025) (decomposition), and AttriBoT (Liu et al., 2025) and CAMAB (Pan et al., 2026a) (perturbation). FlashTrace is run with one recursion and IFR through the token-wise path of the same implementation; IG uses adaptive Simpson integration in fp32. AttriBoT is run as its lossless prefix-cached token leave-one-out component, without the hierarchical, proxy-model and pruning approximations of the original; CAMAB uses linear Thompson sampling for 40 rounds. Implementation sources and adaptations are listed in Appendix B.

Data. The four task families comprise NIAH (100 short-needle and 100 long-needle variants sharing source problems), multi-hop QA over fictional passages (100), math problems with an injected misleading reasoning prefix (80: 40 original/counterfactual pairs), and counterfactual multiple choice in which exactly one value token flips the answer (100: 50 pairs), giving 480 inputs from 290 source problems. NIAH, multi-hop and MCQ are synthetic diagnostics, so we call their labels *reference-evidence annotations* rather than human annotations. Of the 473 inputs for which every method produced an attribution, 403 have a valid fixed target for faithfulness (NIAH 179, multi-hop 51, math 79, MCQ 94; wrong answers are excluded from faithfulness but not from localization), 433 have applicable annotations, and 363 have both. Each metric is reported on its own cohort with its n; the joint cohort is used wherever two axes are compared on the same inputs. Each family follows the format of a published benchmark: NIAH-style long-context retrieval with RULER-style haystacks (Hsieh et al., 2024), bridging and comparison multi-hop QA in the format of HotpotQA (Yang et al., 2018) and 2WikiMultiHopQA (Ho et al., 2020), math word problems whose difficulty follows the MATH scale (Hendrycks et al., 2021b), and four-option counterfactual multiple choice in the format of MMLU (Hendrycks et al., 2021a); all four families were generated for this release except the math family, which is the `math-attr` v2 release copied verbatim.

Annotations. Reference evidence is released as span sets over the input, one description per family. NIAH marks the needle sentence in full and, separately, the value inside it, so the same input supports a segment-level and a token-level reading; the pool varies needle length (11–23 versus 53–65 proxy tokens on matched haystacks), needle kind (number, identifier or word), haystack style (prose or noise), insertion depth (0.1 to 0.9 of the document) and the presence of distractor values. Multi-hop marks the two or three support sentences, the bridging entity and the answer token, over eight fictional passages per context. Math keeps the annotations of the source problems unchanged, with the source span of the injected prefix and a hidden unit decomposition separating the numerical-variation and condition-alteration subsets. Counterfactual multiple choice marks the decisive value tokens and records the edit that separates the two variants. Annotations never enter attribution: the evaluator is kept out of the attribution path, no method sees two variants of one problem together, and all variants of a problem stay in one split.

Protocol. The target is scored in batches of four with replay checks. Time is attribution only (search plus decoding for TEMPO), excluding generation and evaluation, measured on shared hardware. Macro averages give the four tasks equal weight, pooling the two NIAH variants. Confidence intervals are 95% percentile intervals from a paired bootstrap (2,000 replicates) that resamples source problems, stratified by task, so paired variants are never treated as independent; each replicate recomputes both methods on the same resampled inputs, and the intervals describe sampling uncertainty for the selected decoder. The paired macro AUC difference against the remaining baselines is IG $+3.37$ $[3.01, 3.74]$, DePASS $+4.08$ $[3.62, 4.52]$, FlashTrace $+4.39$ $[3.95, 4.86]$ and CAMAB $+4.36$ $[3.94, 4.77]$, the same cohorts and construction as the AttriBoT differences reported in Section 4.2. Against AttriBoT the macro Drop@10/20/30% values are 55.8%, 89.4% and 74.4% higher. Search stability is reported separately over four search seeds (Appendix C.6). The *extension* set of Appendix C.5 contains the 335 faithfulness inputs from the 240 source problems not used to select the decoder. The Time column of Table 1 reports attribution time only, measured on all 473 inputs on which every method produced an attribution rather than on the faithfulness cohort. The tables bold the best value in each column and leave a column unbolded when its best value is attained by more than one method, and macro cells come from the unrounded per-task values, so at printed precision a macro cell need not equal the mean of its column.

E METRIC DEFINITIONS

Faithfulness. Let C be the total token cost of the intervention units U of Section 3.1, with $C = \sum_i c_i$, and let $\beta_1 < \cdots < \beta_{20}$ be the budget grid below, each β_j a fraction of C. Write $\beta_0 = 0$ for the empty-set baseline and $P_\pi(\beta_j C)$ for the strict prefix of Section 3.1 at that budget. With $\mathrm{cost}(S) = \sum_{i \in S} c_i$ the token cost of a unit set $S \subseteq U$, and with $\rho_0 = 0$ and $\rho_j = \mathrm{cost}(P_\pi(\beta_j C))/C$ the fraction actually deleted, the signed-drop AUC is the trapezoid area

$$\mathrm{AUC}(\pi) = \sum_{j=1}^{20} (\rho_j - \rho_{j-1}) \, \frac{F(P_\pi(\beta_{j-1} C)) + F(P_\pi(\beta_j C))}{2}, \tag{13}$$

and $\mathrm{Drop@}k\% = F(P_\pi(kC/100))$. Both are normalized per target token by Equation (1), so they are comparable across inputs of one model but not across models (Edin et al., 2025). We hold the replacement operator fixed across all methods and interpret scores comparatively, since deletion scores are known to depend on the operator and to place the model on out-of-distribution inputs (Hooker et al., 2019; Hase et al., 2021; Basu & Chakraborty, 2026).

Budget grid. The 20 non-zero budgets are 0.1, 0.2, 0.4, 0.8, 1.5, 2.5, 4, 6, 8, 10, 15, 20, 25, 30, 40, 50, 60, 70, 85 and 100 percent of C. Each budget is therefore a fraction of the total token cost of the intervention units, the same object as the searched budget $b_0 = \lfloor 0.2\, C \rfloor$ of Section 3.2. The strict-prefix rule of Section 3.1 takes units from the front of the ranking until the next unit would carry the deleted cost past that fraction, so a budget can be left partly unspent when an indivisible atom does not fit, and the realized fraction ρ_j is the cumulative token cost of the selected prefix divided by C, the abscissa in Equation (13). Duplicate deletion sets across budgets or methods are scored once.

Span-balanced α-NDCG. For non-overlapping evidence spans $E_1, \ldots, E_m$ inside the candidate region (the perturbable part of the input, that is, the union of the intervention units U of Section 3.1), define the coverage of a selected set S as

$$G_\alpha(S) = \frac{1}{m} \sum_{j=1}^{m} \frac{1 - (1-\alpha)^{|S \cap E_j|}}{1 - (1-\alpha)^{|E_j|}}, \tag{14}$$

so every span contributes at most $1/m$ and repeated hits inside one span have geometrically decreasing gain. With S_r the first r units of the ranking, $N = |U|$ the number of ranked units, and $g_r = G_\alpha(S_r) - G_\alpha(S_{r-1})$,

$$\mathrm{SB}\alpha\text{-NDCG@}k = \frac{\sum_{r=1}^{\min(k,N)} g_r / \log_2(r+1)}{\mathrm{IDCG}_\alpha @k}, \tag{15}$$

where the ideal DCG merges the per-span geometric gain sequences in decreasing order, which is exact when spans do not overlap. We use $\alpha = 0.5$ and $k = 10$ units; k counts ranked units, while the deletion budget of Equation (13) is a separate quantity. For MCQ, whose variants are reported in Appendix C.4, the ideal ordering may be unattainable under the atomic-unit constraint, so the metric is an upper-bounded localization score. Recall@20% divides by all reference tokens, including any that fall outside the candidate region. The localization metrics measure *plausibility* (Jacovi & Goldberg, 2020), not faithfulness, and the two axes are kept separate throughout.

E.1 INFORMATION LOSS AND ATTRIBUTION COMPATIBILITY IN AUC METRICS

Our choice of the primary faithfulness metric is motivated by two diagnostic findings: the normalized AUC we tested frequently discards observed response information, while MAS additionally evaluates compatibility between attribution magnitudes and a particular deletion trajectory.

Clipping and irreversible suppression in normalized AUC. Let L_t be the fixed target's log-probability after deleting a ranked prefix containing a fraction ρ_t of eligible tokens. For inputs with $L_0 > L_T$, the normalized response is

$$z_t = \frac{L_t - L_T}{L_0 - L_T}, \qquad \bar{z}_t = \mathrm{clip}_{[0,1]}(z_t), \qquad R_t = \min_{0 \le s \le t} \bar{z}_s. \tag{16}$$

The reported normalized AUC integrates R_t. This transformation discards information in two distinct ways. First, clipping maps all responses below the full-deletion reference to zero and all responses above the original reference to one, removing their magnitudes. Second, the running minimum suppresses subsequent response increases; in particular $L_\tau \leq L_T$ forces $R_t = 0$ for every $t \geq \tau$, so once that threshold is crossed later observed responses no longer affect the normalized AUC, even if the model substantially recovers.

We audited these operations using the frozen curves of all 11 methods, evaluated at 20 nonzero deletion budgets plus the undeleted baseline. The audit includes 400 inputs with positive endpoint gaps; three MCQ inputs with nonpositive gaps are excluded. For each input and method, we counted distinct interior deletion masks, excluding both endpoints: budgets that delete no token collapse onto the same mask, so the proportions below are taken over distinct interior masks rather than over budget indices. We measured

$$p_- = \Pr(z_t < 0), \qquad p_+ = \Pr(z_t > 1), \qquad p_{\text{sup}} = \Pr(R_t < \bar{z}_t), \tag{17}$$

where the probabilities are taken over the distinct interior masks of one input and method pair. Table 18 averages these proportions first over inputs within each task and then equally over methods.

Table 18: Information discarded by the tested normalization pipeline. The three percentages describe sampled interior masks; suppression can overlap with clipping. A_{clip} integrates $\bar{z}_t$, whereas A_{norm} additionally applies the running minimum. All areas use actual deletion fractions.

Task	n	p_-	p_+	p_{sup}	A_{clip}	A_{norm}
NIAH	179	75.4%	1.4%	9.0%	0.02609	0.02170
Multi-hop	51	48.9%	3.7%	28.1%	0.16517	0.08483
Math	79	4.6%	2.6%	25.7%	0.30903	0.29229
MCQ	91	49.1%	3.4%	17.6%	0.31997	0.20668

Clipping is especially frequent on NIAH: across individual methods, between 37.6% and 90.8% of interior responses fall below the full-deletion reference. For FlashTrace and the free-seed control, respectively, 89.9% and 93.9% of NIAH inputs reach the absorbing zero response by a deletion budget of only 1%; their resulting normalized AUCs are 0.00130 and 0.00158. Such near-zero values compress substantially different observed responses into almost identical saturated curves. The running minimum also removes meaningful recoveries: for the search-only ordering on multi-hop, 47.1% of inputs recover to $\bar{z}_t > 0.1$ after first reaching zero, and these recoveries are entirely hidden by R_t; its clip-only AUC is 0.06793, against 0.02632 after the running minimum, and on MCQ the corresponding values are 0.10426 and 0.02607. These findings concern clipping and monotonicization rather than endpoint normalization alone: removing only the running minimum cannot recover the magnitudes already discarded by clipping.

MAS imposes an additional preference on score representation. MAS (Walker et al., 2024) inherits the processed response R_t and compares it with the remaining attribution mass,

$$D_t = 1 - \frac{\sum_{i \in S_t} w_i}{\sum_i w_i}, \qquad C_t = \text{clip}_{[0,1]}\left(R_t + |R_t - D_t|\right), \tag{18}$$

before a final rescaling and integration. Thus, even with the deletion ordering and the model responses held fixed, MAS depends on how attribution magnitudes are represented. For a fixed response, the alignment penalty is minimized by $D_t = R_t$: the preferred magnitudes encode the processed response along this particular deletion path. This is an additional calibration requirement on the representation of attribution magnitudes; it constrains that representation in addition to ordering quality. The interaction with saturation makes the distinction clear: for nonnegative weights with positive total mass, $R_t = 0$ implies $C_t = D_t$, so after response saturation the corrected curve depends on remaining attribution mass while ignoring subsequent model-response differences. MAS therefore favors scores whose mass is concentrated consistently with the retained response decreases. Our sensitivity tests show that this mismatch materially affects evaluation: keeping every deletion mask and model response unchanged, replacing signed weights by their positive parts solely for the MAS mass calculation changes the four-task macro MAS of IG from 0.5995 to 0.2112 and that of Grad×Input from 0.7743 to 0.4046 (lower is better). Moreover, IG's original remaining-mass curve leaves $[0, 1]$ on $376/403$ inputs, while CAMAB triggers the nonpositive-total-mass fallback on all 403 inputs. These changes and fallbacks concern score compatibility, not improved deletion rankings: they establish a representation preference rather than a universal advantage for any attribution family.

Implication for the primary metric. We consequently use the signed-drop AUC of Equation (13), that is

$$\mathrm{AUC}_{\mathrm{SD}} = \sum_{t=1}^{T} (\rho_t - \rho_{t-1}) \frac{(L_0 - L_{t-1}) + (L_0 - L_t)}{2m}, \tag{19}$$

where m is the fixed target length, so the score is $F = (L_0 - L_t)/m$ in the notation of Equation (13). It retains observed decreases, negative drops and recoveries without requiring attribution scores to form a calibrated mass decomposition.

F Full Lineage and Secondary Comparisons

This section carries the classification of existing methods, the complete research lineage and the secondary comparisons behind Section 1 and Section 2; the nearest-neighbour relations and the differences that decide the contribution of TEMPO are in the body.

Method families. Existing methods fall into families distinguished by what they read from the model. *Information-flow* and *decomposition* methods read the model's internal routes and are the cheapest, but each is defined over the softmax-attention forward pass (Ferrando & Voita, 2024; Pan et al., 2026b; Hong et al., 2025). *Perturbation* methods need only scoring access and estimate exactly what faithfulness evaluation measures, the change in the response's log-probability when a source is removed (Cohen-Wang et al., 2024; Trinh et al., 2026), at a cost growing with the number of sources.

Model-coupled methods. Model-coupled methods read internal quantities: information flow aggregates per-layer attention and feed-forward contributions into token importance (Abnar & Zuidema, 2020; Kobayashi et al., 2020; Ferrando et al., 2022; Ferrando & Voita, 2024), FlashTrace extends this to multi-token targets and recursion through reasoning chains (Pan et al., 2026b), relevance propagation adds attention-specific rules (Achtibat et al., 2024), decomposition propagates additive components through a fixed forward pass (Hong et al., 2025), and gradient and Shapley estimators are architecture-agnostic in principle but need gradients or many evaluations at LLM scale (Sundararajan et al., 2017; Lundberg & Lee, 2017; Barkan et al., 2021; Zhao et al., 2024; Shapley, 1953; Xiao et al., 2025; Enouen et al., 2023). Mamba-1 admits an implicit-attention reformulation and dedicated relevance rules (Ali et al., 2024; Rezaei Jafari et al., 2024), but we know of no extension of them to gated delta-rule layers or the interleaved hybrids deploying them, where hybrid-model interpretability relies on component ablation and cache-level interventions (Borobia et al., 2026; Afendulev et al., 2026), and in our reproduction the pinned FlashTrace value path fails to reconstruct a Qwen3.5 mixer output (Appendix B).

Black-box and mask-search methods. Black-box methods need scoring access only: ContextCite fits a sparse surrogate to random ablations (Cohen-Wang et al., 2024), AttriBoT approximates leave-one-out with cached prefixes and proxies (Liu et al., 2025), CAMAB selects subsets by Thompson sampling (Pan et al., 2026a), TracLLM searches long contexts (Wang et al., 2025), and ReAGent and MExGen extend perturbation to tokens and to text-only APIs (Zhao & Shan, 2024; Paes et al., 2025). Mask optimization against a black-box score has a long history in vision: learned sparse or fixed-area masks (Fong & Vedaldi, 2017; Fong et al., 2019), random-mask averaging (Petsiuk et al., 2018), superpixel subset selection by greedy, phase-window or cross-entropy search (Chen et al., 2024; 2025; Gu et al., 2025; 2026), unified as removal-based explanations by Covert et al. (2021); in text, learned rationales (Lei et al., 2016; Bastings et al., 2019) or k-sparse local search (Hase et al., 2021). Word and clause boundaries delimit the local expressions carrying a dependence, and explanation methods for classifiers have long used this structure to *score* or *organize* spans (Lei et al., 2016; Chen et al., 2020; Jin et al., 2020; Tsang et al., 2020), while prior mask-search work obtains a complete budget-nested order only through a further model-querying step (Hase et al., 2021; Gu et al., 2026).

Evaluating attribution. Deletion-based faithfulness metrics (AOPC, comprehensiveness and sufficiency, insertion/deletion AUC) are standard (Nguyen, 2018; DeYoung et al., 2020; Petsiuk et al., 2018; Chan et al., 2022), with top-k log-probability drop the established form for LLM context attribution (Cohen-Wang et al., 2024; Liu et al., 2025); their pitfalls are documented: out-of-distribution inputs (Hooker et al., 2019; Hase et al., 2021; Madsen et al., 2022a), non-comparable bounds across models (Edin et al., 2025), operator dependence (Kamahi & Yaghoobzadeh, 2024; Basu &

Chakraborty, 2026) and magnitude blindness (Zhao & Aletras, 2023; Walker et al., 2024). Agreement with human or reference rationales is a separate criterion (DeYoung et al., 2020; Jacovi & Goldberg, 2020; Ju et al., 2022), known to rank methods differently in classification (Nguyen, 2018; Atanasova et al., 2020; Brandl & Eberle, 2026; Banerjee et al., 2026); broader concerns about attribution and self-explanation reliability in LLMs are surveyed elsewhere (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019; Bastings & Filippova, 2020; Madsen et al., 2022b; Lyu et al., 2024; Lanham et al., 2023; Turpin et al., 2023).