A NARROW RESIDUAL BOTTLENECK FROM EVIDENCE TO ANSWER IN LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

When a language model answers a question about a passage, the answer hinges on a small span of the input. We ask where inside the model that information is forced to pass, and find a *narrow residual bottleneck*: a small set of evidence-token positions at a single residual-stream layer that carries the evidence-to-answer computation on its own. We locate it with two hidden-state interventions on a single candidate set. One copies the original evidence into a corrupted run and asks whether the correct answer returns (sufficiency); the other injects the corrupted state into a clean run and asks whether the answer breaks (necessity). Requiring the same set to pass both is strictly stronger than either direction alone. Against a size-matched permutation null, the existence of a bottleneck becomes falsifiable: a bottleneck must pass the two-sided test far more often than random sets of its own size, and it does. On the controlled benchmark, in our best-powered models the cut passes about 0.98 of the time against about 0.01 for the matched control, and is narrow (typically two to three positions, one to two on real text) throughout. It reappears across three normalization schemes on real single-hop, multi-hop, arithmetic, and long-context question answering, and at a fixed budget steers the answer far more reliably than editing a random set. Most tellingly, the bottleneck is not simply the top of an importance ranking: fixing the set size, the test, and the model, and changing only how the set is chosen, the same-size set of highest-importance attention heads fails the very test the located cut passes, a gap that holds across four model families.

1 INTRODUCTION

When a language model reads a passage and answers a question about it, the answer depends on a small part of the input: the span of text that holds the evidence. A large body of interpretability work asks which components matter for such behaviors, scoring each head, edge, or feature by how much it contributes (Conmy et al., 2023; Syed et al., 2023; Marks et al., 2025; Wang et al., 2023a; Wu et al., 2025; Olsson et al., 2022). We ask a different question. Instead of ranking components by importance, we look for the bottleneck: the smallest set of internal positions the evidence must pass through to reach the answer, which we call the cut. Importance and bottleneck are not the same object: many components can each be important without any small set of them forming a neck that all the information funnels through. We show that in practice they come apart.

Why this study matters. Knowing that a component is important is not the same as knowing an answer's computation runs through it. Section 4.4 will show the difference pays off as an edit target: the verified cut beats an importance-ranked edit at the same budget. The same gap reaches further than editing: any audit that uses an importance score to certify what a system's output depends on inherits that gap. Section 4.3 will show why that certification falls short: a per-component score, however defined, cannot on its own tell whether the set it selects is, as a whole, both sufficient and necessary. The audit still returns an answer, yet it is just not the audit it claims to be. What it needs first is the object this paper locates: a bottleneck whose existence is checked, not assumed. An importance ranking is a proxy a system has to trust; a falsifiable bottleneck is a claim it can check before it builds on it.

Contribution. We study a *narrow residual bottleneck*: a small set of evidence-token positions, at a single residual-stream layer, that carries the evidence-to-answer computation on its own. We

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.

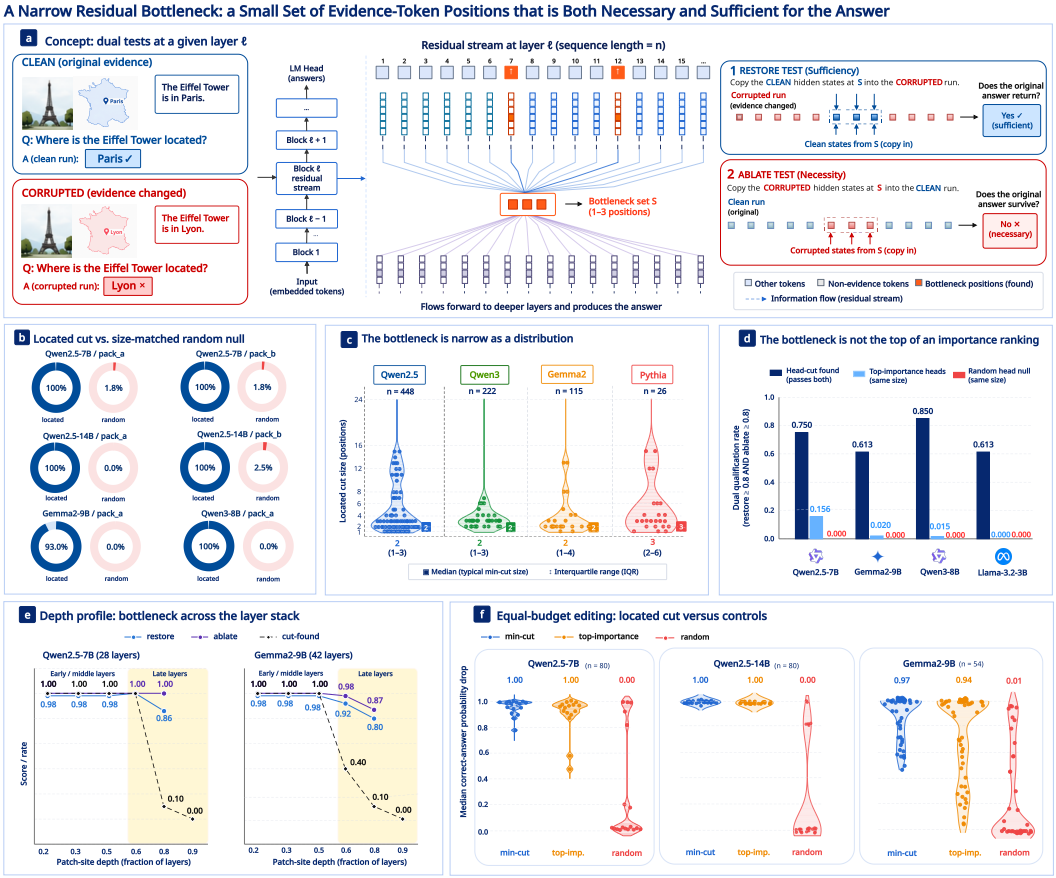

Figure 1: **A narrow residual bottleneck: necessary and sufficient for the answer. (a)** At a chosen residual-stream layer ℓ, we search for the smallest position set S that both restores the answer in a corrupted run (sufficiency) and severs it in a clean run (necessity). **(b)** The located cut passes this dual test in 0.93–1.00 of cases, against a size-matched random control at 0.000–0.025 ($\approx 107\times$ separation; representative cells, full seven in Table 1). **(c)** The neck is narrow as a distribution, not just on average, pooled per family across this work's experiments; Pythia (fifth-architecture coverage point, no scaling claim) is wider. **(d)** The cut is not recoverable from a per-head importance ranking, defined and explained in Section 4.3: a same-size top-importance-head set rarely passes, and a random head-set never does, across four families (Table 3). **(e)** The cut spans early-to-middle layers and disappears late, on Qwen2.5-7B and Gemma2-9B (Section 4.6). **(f)** At equal edit budget, patching only the located cut drops correct-answer probability at least as much as a top-importance set and far more than a random set (median drop; break-rate per pack separately in Table 4).

make this precise with two interventions on the model's hidden state. To build a controlled test, we take a prompt the model answers correctly and make a scrubbed copy (an edited version in which the evidence is changed so the correct answer flips). Given a candidate set of positions, a restore intervention copies the original hidden state at those positions into the scrubbed run and asks whether the correct answer returns (sufficiency); an ablate intervention copies the scrubbed state into the original run and asks whether the answer breaks (necessity). Our estimand is the smallest set passing both interventions at once, recovered by greedy search: its size upper-bounds that of any truly minimal set, and an exhaustive check finds the bound tight in $\approx 95\%$ of cases (Section 4.1). Section 3 states all of this formally. Each element of this test has a counterpart in prior work; what sets this test apart is detailed in Section 2.

Concretely, we contribute:

1. **The dual residual bottleneck.** The smallest set of evidence-token positions, at a single residual-stream layer, that both restores a scrubbed prompt and severs a clean one. Re-

quiring one set to pass both directions is stricter than one-sided faithfulness or the "no redundant head" minimality of prior circuit work.

2. **A falsifiable test for a true bottleneck.** The size-matched permutation null turns "a narrow neck seems to exist" into a claim that can fail: the located cut must pass both interventions far more often than random sets of the same size, and it does so by nearly two orders of magnitude (Appendix F.2). This control is absent from most circuit-faithfulness evaluations (Miller et al., 2024).
3. **The cut generalizes and steers.** The narrow, control-beating cut reappears on real single-hop, multi-hop, arithmetic, and long-context question answering, and at a fixed editing budget, patching only the cut steers the answer far more reliably than patching a random set of the same size.
4. **A bottleneck is not the top of an importance ranking.** Holding the set size, the two-sided test, and the model fixed and changing only how the set is selected, the same-size set of highest-importance attention heads fails the two-sided test that the located cut passes, across four model families, with a random-head control passing in exactly none. Only a search for a cut recovers a set that qualifies; importance does not. How importance is defined and why the two measures diverge are discussed in Section 4.3 (case details in Appendix G.1), and how prior work defines the same criterion is in Appendix A.

2 Related Work

Circuit-discovery methods (ACDC (Conmy et al., 2023), EAP (Syed et al., 2023), SFC (Marks et al., 2025), IOI (Wang et al., 2023a)) and in-context localization work (retrieval heads (Wu et al., 2025), function vectors (Todd et al., 2024), induction heads (Olsson et al., 2022)) each rank or select components by a single importance criterion; we instead search for a bottleneck over token positions under a two-sided necessity-and-sufficiency test, calibrated against a size-matched permutation null. Though the exact formula varies (edge or feature ablation, a gradient approximation of it, or head-level activation score), each reduces to one scalar per component, not a set tested jointly against a null. Section 4.3's baseline (direct logit attribution) is the same kind of measure, held fixed against the located cut so the dissociation is not an artifact of a weak baseline. Heimersheim & Nanda (2024) show restore and ablate scores are "not complements" but do not require one set to pass both; Miller et al. (2024) show faithfulness is sensitive to the ablation operator, a sensitivity orthogonal to our null (Appendix L). Circuit-size scaling on Pythia (Tigges et al., 2024) measures component count, not neck width; we use Pythia only as a coverage point (Appendix K.2). ROME/MEMIT (Meng et al., 2022; 2023) localize stored facts by denoising alone; our downstream-utility result (Table 4) instead targets an in-context object. Each element above has a neighbor in this prior work, but none requires one located set to clear both directions of the test against a size-matched null, a comparison that makes the importance/bottleneck dissociation (Section 4.3) testable at all. A full discussion of related work, including causal scrubbing (Chan et al., 2022), label-word anchors (Wang et al., 2023b), and how previous work operationalizes importance, is in Appendix A.

3 Methodology

3.1 Setup and notation

Let M be a language model with D transformer blocks. Given a prompt x, write $h^{(L)}(x) \in \mathbb{R}^{T\times d}$ for the residual stream entering block $L{+}1$, where T is the number of tokens and d the model width; row $h_t^{(L)}(x)$ is the hidden state at token position t. The model emits a distribution over a fixed label set $\mathcal{Y}$, and we write $p_M(y \mid x)$ for the probability of label y.

An evidence-to-answer task presents x with N candidate evidence spans and a rule that selects one of them; the correct label $y^\star(x)$ is determined by the selected span. For each prompt we construct a scrubbed counterfactual $\tilde{x}$ by replacing the selected span with a competing one, so that the correct label changes, $y^\star(\tilde{x}) \neq y^\star(x)$, while everything else is held fixed. This pairing is what makes the interventions below causal: x and $\tilde{x}$ differ only in the evidence, so any change in the answer we can induce by patching a set of positions is attributable to those positions.

We restrict analysis to prompts the model actually solves and where the counterfactual is informative. Writing $\Delta(x) = p_M\big(y^\star(x) \mid x\big) - p_M\big(y^\star(x) \mid \tilde{x}\big)$ for the drop in correct-label probability caused by scrubbing, a prompt is admissible if $\arg\max_y p_M(y \mid x) = y^\star(x)$ and $\Delta(x) \geq \tau_{\text{harm}}$. The threshold τ_{harm} excludes prompts that are unsolved or whose counterfactual barely moves the answer; it also lower-bounds the denominator $\Delta(x)$ used to normalize the scores below, so normalization is well-conditioned on every admissible prompt. Its value and constants are in Appendix E.

3.2 The intervention site

All interventions act on the inter-block residual $h^{(L)}$: the tensor entering block $L{+}1$ after every normalization inside block L. We fix L to the middle of the network a priori and never tune it to the result (no per-prompt tuning of L; Section 4.6 later shows the cut holds across early-to-middle depths). This site is defined identically regardless of a model's normalization scheme (pre-, sandwich-, or post-norm) or its use of query/key normalization, so a cross-family comparison is not confounded by where normalization is applied (Section 4.2 verifies this to machine precision).

3.3 Restore, ablate, and the dual cut

Given a set of positions $S \subseteq \{1, \ldots, T\}$, define the patch operator $\big(h \to_S h'\big)$ that overwrites the rows of h indexed by S with the corresponding rows of h' and leaves the rest unchanged:

$$\big(h \to_S h'\big)_t \;=\; \begin{cases} h'_t & t \in S, \\ h_t & t \notin S. \end{cases} \tag{1}$$

Let $q_M(y \mid x; h)$ denote the model's label probability when its residual at layer L is set to h. The two interventions measure the resulting change in correct-label probability, normalized by the causal gap $\Delta(x)$:

$$\text{restore}(S) \;=\; \frac{1}{\Delta(x)}\Big[\, q_M\big(y^\star(x) \,\big|\, \tilde{x};\, h^{(L)}(\tilde{x}) \to_S h^{(L)}(x)\big) \;-\; p_M\big(y^\star(x) \mid \tilde{x}\big)\Big], \tag{2}$$

$$\text{ablate}(S) \;=\; \frac{1}{\Delta(x)}\Big[\, p_M\big(y^\star(x) \mid x\big) \;-\; q_M\big(y^\star(x) \,\big|\, x;\, h^{(L)}(x) \to_S h^{(L)}(\tilde{x})\big)\Big]. \tag{3}$$

Both scores are normalized against the causal gap: patching $S = \varnothing$ gives 0, and patching the full evidence region gives 1, so each score is the fraction of the evidence's effect that the patched positions reproduce. The scores are not confined to $[0, 1]$: a patch that drives the probability past the opposite reference run yields a value beyond that range. The threshold θ below reads as "recovers at least a θ fraction of the causal gap" rather than as a bounded score. $\text{restore}(S)$ measures *sufficiency*: it copies the original evidence-carrying state into the scrubbed run at S and asks how much of the correct answer returns. $\text{ablate}(S)$ measures *necessity*: it copies the scrubbed state into the original run at S and asks how much of the correct answer is lost. A set S dual-qualifies at threshold θ if it passes both directions and the argmax flips as expected under each:

$$\text{dual}_\theta(S) \;=\; \big[\text{restore}(S) \geq \theta\big] \;\wedge\; \big[\text{ablate}(S) \geq \theta\big] \;\wedge\; \big[\hat{y}_{\text{res}}(S) = y^\star(x)\big] \;\wedge\; \big[\hat{y}_{\text{abl}}(S) \neq y^\star(x)\big], \tag{4}$$

where $\hat{y}_{\text{res}}(S)$ and $\hat{y}_{\text{abl}}(S)$ are the argmax labels after the restore and ablate patches. The two argmax conditions guard against a patch that moves probability mass without flipping the decision. Requiring the same set to pass both directions is strictly stronger than either alone: a set can be sufficient without being necessary, or the reverse, and only their intersection identifies a true bottleneck (Heimersheim & Nanda, 2024) (illustrated in Figure 1a).

3.4 The causal cut and its estimand

We define the minimal causal cut as the smallest dual-qualifying set drawn from a candidate pool P of evidence-token positions:

$$C^\star \;=\; \arg\min_{S \subseteq P} |S| \quad \text{subject to} \quad \text{dual}_\theta(S). \tag{5}$$

$C^\star$ is our estimand. Exhaustive search over all subsets is exponential, so we do not compute $C^\star$ directly; instead we recover a dual-qualifying set $\hat{C}$ greedily, starting from $S = \varnothing$ and repeatedly

adding the position that most increases the weaker of the two scores,

$$S \leftarrow S \cup \Big\{ \arg\max_{p \in P \setminus S} \min\big(\text{restore}(S \cup \{p\}),\, \text{ablate}(S \cup \{p\})\big) \Big\}, \tag{6}$$

stopping as soon as $\text{dual}_\theta(S)$ holds, and setting $\hat{C} = S$. We report $|\hat{C}|$, or record "no cut" when no subset of P dual-qualifies. Because greedy search can overshoot when a cut requires two positions that are useful only jointly, $|\hat{C}| \geq |C^\star|$: the reported size is an upper bound on the estimand. Section 4.1 measures how tight this bound is by an exhaustive check on small cuts, finding $\hat{C} = C^\star$ in $\approx 95\%$ of cases. When no cut is found we report that as an outcome rather than dropping the prompt.

3.5 The size-matched permutation null

A small cut is meaningful only if random small sets do not pass the same test. For a located cut of size $k = |\hat{C}|$, the null qualification rate is the probability that a random size-k subset of a pool Q also dual-qualifies:

$$\rho_{\text{null}}(Q, k) \;=\; \Pr_{S \sim \text{Unif}\binom{Q}{k}} \big[\text{dual}_\theta(S)\big], \tag{7}$$

estimated by sampling n_{null} size-k sets per case. Per-case rates are therefore resolved to $1/n_{\text{null}}$, and we report n_{null} alongside every null rate; a per-cell rate of 0.000 means no draw qualified at that resolution. Aggregating across cases resolves the null far more tightly than any single case allows (Section 4.1). We use two pools: the relevant pool (the same evidence positions the cut is drawn from, the hardest baseline since these positions are "important on paper") and the all-evidence pool, which also includes competing spans. The cut is a true bottleneck when ρ_{null} falls far below the located cut's qualification rate, so that the existence of a narrow neck is a falsifiable claim, not an assertion. Head-level random controls (Wu et al., 2025) test a similar intuition at a coarser grain; the size-matched positional null is what most circuit-faithfulness evaluations lack.

3.6 What we evaluate

We run the same pipeline (admissibility filter, cut search, and permutation null) across several settings. Beyond a controlled routing benchmark, we test four real question-answering tasks: single-hop (SQuAD; Rajpurkar et al., 2016), multi-hop (HotpotQA; Yang et al., 2018), arithmetic (GSM8K; Cobbe et al., 2021), and long-context (Natural Questions; Kwiatkowski et al., 2019). For real text we enlarge the null pool to include uncontaminated tokens before the answer span, since answer-span positions alone sit so close to the cut that random draws would nearly reproduce it. We measure downstream utility by comparing, at an equal editing budget, how often patching only the cut breaks the answer against patching a top-importance set or a random set of the same size. Finally, we test whether the cut is recoverable from an importance ranking: where a cut exists over individual attention heads, we check whether the same-size set of highest-importance heads passes the dual test. Full protocol details are in Appendices C and D.

4 Experiments

We organize results around six questions. **(Q1)** Is the located cut narrow, and does it beat a size-matched permutation null? **(Q2)** Does the site validate across normalization geometries? **(Q3)** Is the cut recoverable from a per-head importance ranking? **(Q4)** Does patching only the cut break the answer at equal edit budget? **(Q5)** Does the narrow, null-beating neck replicate on real single-hop, multi-hop, arithmetic, and long-context QA? **(Q6)** Is the bottleneck confined to the fixed layer, or does it span a band of depths? All numbers trace to summary artifacts in the released repository provided in reproducibility statement. Baselines throughout are (i) the size-matched permutation null, (ii) top-importance sets at matched edit budget, and (iii) random-head controls.

4.1 Q1: A narrow, null-beating cut on the controlled benchmark

Core claim. Table 1 reports the position-level cut on seven primary cells (Qwen2.5-7B ×2 packs, Qwen2.5-14B ×2 packs, Gemma2-9B, Qwen3-8B ×2 packs), 60–100 valid cases per cell. The patch

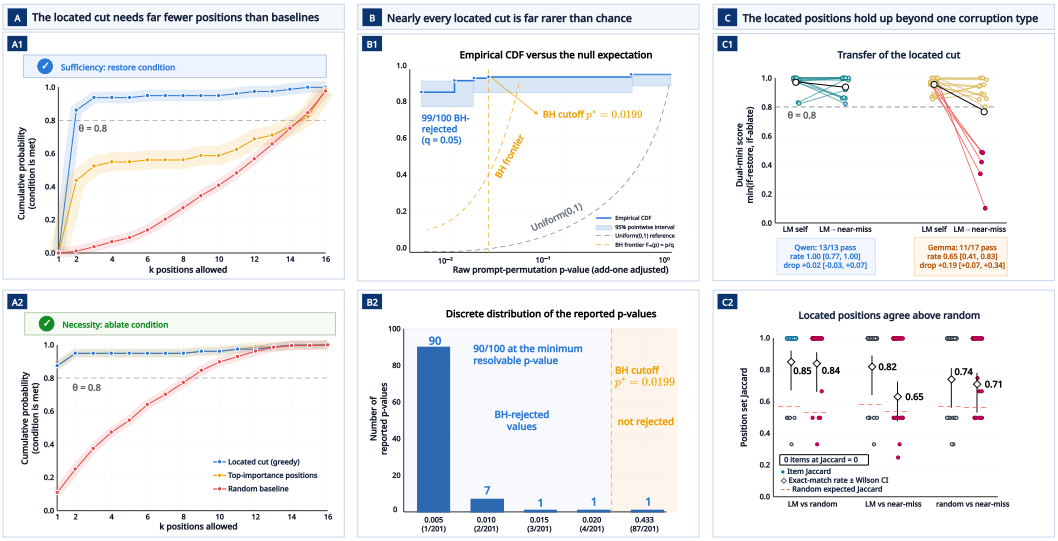

Figure 2: **Three additional checks that stress-test the core claims. (A)** The located cut needs far fewer positions than either baseline: a median of 2 positions reaches sufficiency, versus 3 for a top-importance ordering and 8 for a random ordering, at the same success rate (Appendix G.2). **(B)** The null-beating separation is not a pooling artifact: a per-case permutation test, corrected for testing 100 cases at once (Benjamini–Hochberg (Benjamini & Hochberg, 1995), $q = 0.05$), is individually significant in 97–99% of cases (Appendix F.4). **(C)** The located cut keeps working, not just keeps its coordinates, under a different corruption method: it transfers to the hardest, most confusable corruption in $13/13$ Qwen2.5-7B cases and $11/17$ Gemma2-9B cases (Appendix I.4).

Table 1: **Core result: a narrow, null-beating cut on the controlled benchmark** (position-level). Cut-found 95% CI is Wilson; null 95% CI is a bootstrap over per-case null rates, each estimated from $n_{\text{null}} \in \{10, 12\}$ draws; main text discusses the broader pooled estimate.

Family (norm)	Scale/Pack	n	acc	cut-found [CI]	size	n_{null}	null [CI]
Qwen2.5 (pre)	7B/a	100	0.79	1.00 [.96,1.0]	2.56	12	.018 [.002,.039]
Qwen2.5 (pre)	7B/b	100	0.88	1.00 [.96,1.0]	2.40	12	.018 [.001,.043]
Qwen2.5 (pre)	14B/a	60	0.93	1.00 [.94,1.0]	2.00	10	.000 [.000,.000]
Qwen2.5 (pre)	14B/b	60	0.92	1.00 [.94,1.0]	2.38	10	.025 [.000,.060]
Gemma2 (sandwich)	9B/a	60	0.65	0.93 [.84,.97]	2.62	10	.000 [.000,.000]
Qwen3 (pre)	8B/a	60	0.90	1.00 [.94,1.0]	2.75	10	.000 [.000,.000]
Qwen3 (pre)	8B/b	60	0.82	0.95 [.86,.98]	2.88	10	.004 [.000,.009]

layer is fixed a priori ($L = \lfloor \text{layers}/2 \rfloor$), never tuned per case. The located cut dual-qualifies on almost every case while random size-matched sets almost never do: averaged across these seven cells, the qualification rate is 0.983 against a null of 0.009 (Figure 1b). Mean cut size is 2.0–2.9 positions on a 10-group candidate pool (15–19 token positions).

Not a small-sample artifact. Per-case null rates are individually resolved only to $1/n_{\text{null}}$ with $n_{\text{null}} \in \{10, 12\}$ draws, too coarse alone for a meaningful interval; pooling across a broader set of 636 located-cut cases gives a tight aggregate estimate of 0.011, 95% CI $[0.005, 0.017]$. The all-evidence null, including distractor spans, is more decisive still, 0.000–0.005 in these seven cells (Appendix F.2: 0.000–0.008 pooled over the broader 13-cell set), and this separation holds as a distribution across families, not only on average (Figure 1c).

Threshold and hard-null robustness. Three controls (Appendix F.1) show this is a property of the mechanism, not the definition: cut-found rate and size are stable through $\theta = 0.8$ and decline only gradually to 0.9; the cut still beats an importance-matched hard null (0.055–0.064 at $n_{\text{null}} = 40$ against the top-$2k$ important positions); and an exhaustive check finds the greedy cut equals $C^\star$ in $\approx$95% of cases.

Null-pool and pretraining robustness. The separation also holds regardless of null-pool composition and size (an $11\times$ range across real-data cells), with the strictest control-only null at 0.000

throughout (Appendix F.2), and appears at the same magnitude in the Qwen2.5-7B base model, so it is not solely an artifact of instruction tuning (Appendix F.3).

Not a pooling artifact. One might worry this separation only shows up after pooling many cases together; it does not: Figure 2B shows a per-case significance test, corrected for testing 100 cases at once, still rejects the null in 97–99% of cases individually (Appendix F.4).

4.2 Q2: The site validates across normalization geometries

The intervention site (Section 3.2) should be the same object in every model regardless of where normalization is applied. Table 2 confirms this: across seven families spanning three normalization schemes, the state we read at the site matches the true input to the next block to machine precision (largest difference 0 everywhere). A cross-family comparison is therefore not confounded by normalization placement, and the site cannot silently yield false negatives on models that normalize differently.

Table 2: **The intervention site is the same object across seven families and three normalization schemes.** The last column is the largest difference between the state read at the site and the true input to the next block; 0 means identical to machine precision.

Family	Normalization scheme	post-FF norm	Q/K norm	Largest difference
Qwen2.5	pre-norm	no	no	0.0
Qwen3	pre-norm	no	yes	0.0
Llama-3.2	pre-norm	no	no	0.0
Mistral-v0.3	pre-norm	no	no	0.0
Gemma2	sandwich-norm	yes	no	0.0
Gemma3	sandwich-norm	yes	yes	0.0
OLMo2	post-norm	yes	yes	0.0

4.3 Q3: An importance ranking is not a cut

This is the central test of the paper: whether the cut is anything more than an importance ranking under another name. We run the search at the finer grain of individual attention heads in the later layers and ask, holding the set size and the two-sided test fixed and changing only how the set is chosen: when a cut over heads exists, does the same-size set of *highest-importance* heads also pass? Importance at the head level is direct logit attribution to the decision margin, a head's output projected onto the unembedding direction for the correct label minus its best competitor, read directly off the clean forward pass (Appendix C). **It does not pass**, and the dissociation holds across four families spanning two normalization geometries (Table 3). In every family a head-cut is found in a majority of cases, yet the same-size top-importance-head set passes the two-sided test almost never (0.00–0.16), and a random head-set never does (0.000 at $n_{\text{null}} = 40$ in all cells; Figure 1d).

Table 3: **An importance ranking is not a cut, across four families.** Head-level search in a late-layer band. When a head-cut is located, the same-size set of highest-importance heads almost never reproduces it and a random head-set never does ($n_{\text{null}} = 40$ draws per case). Case selection and per-pack detail in Appendix G.1; the main text explains why the two measures diverge.

Model	Family (norm)	n	head-cut found	top-imp. qualifies	random-head null
Qwen2.5-7B	Qwen2.5 (pre)	60	0.750	0.156	**0.000**
Gemma2-9B	Gemma2 (sandwich)	80	0.613	0.020	**0.000**
Qwen3-8B	Qwen3 (pre, QK-norm)	80	0.850	0.015	**0.000**
Llama-3.2-3B	Llama3.2 (pre)	80	0.613	0.000	**0.000**

Why the two measures disagree. A bottleneck is, in one sense, itself a kind of importance: a claim about what the decision cannot do without. What differs is how that claim is tested. Direct logit attribution is a linear read of one clean run, not a causal quantity, so it cannot see a head whose role only appears once another head is disrupted, such as a backup head that compensates for an ablated partner (Wang et al., 2023a). More generally, any per-component ranking, causal or not, scores components one at a time and never checks whether a specific set clears a joint necessity-and-sufficiency test against a null of equally sized alternatives; a component can rank low individually yet belong to the cut, or rank high yet be replaceable, and only a joint test can tell the two apart.

Two scoping points make the claim precise. First, head-cut existence is model-dependent (found-rate 0.61–0.85) and wider-head architectures route through wider head-necks ($\approx$ 5–6 heads for Qwen2.5/Llama vs. $\approx$ 8–9 for Gemma2/Qwen3), which is why we state narrowness at the position level and dissociation at the head level, where it is clean; the dissociation still holds at low accuracy (Llama-3.2-3B, 0.40 task accuracy, head-cut found 61% of the time, top-importance qualifying at exactly 0.00), ruling out a large-capable-model explanation (Appendix G.1). **Second,** position-level importance passes more often (about 0.6) because it is already a causal ablation score, closer in kind to the dual test than head-level DLA, though it remains a single-position score rather than a joint set test and so still falls short of the cut. This is also why the dissociation is sharpest exactly where circuit-discovery methods like path patching (Goldowsky-Dill et al., 2023), ACDC, and EAP operate (SFC targets a finer-grained decomposition into sparse features): at head-level granularity, neither the causal nor the joint-test cushion applies. The gap holds across budgets, not just at the stopping size: the cut reaches sufficiency at a median of two positions against three for importance and eight for random (Figure 2A; Appendix G.2).

Why this matters for editing. At the position level, where importance is the better-behaved of the two measures above, Section 4.4 will show the cut beats a top-importance edit at equal budget; the head-level dissociation here suggests an importance-ranked edit would fare worse still at the finer granularity most editing and steering methods would need.

4.4 Q4: The cut is an effective causal edit target

At equal edit budget, patching only the located-cut positions breaks the answer 100% of the time across six (family, pack) cells (214 steering cases), against 20–53% for a size-matched random relevant set (1.9–5.0× separation; Table 4, Figure 1f). A top-importance set is a competitive heuristic (59–100%) but never better than the dual-cut search, and clearly worse on Gemma2-9B (0.86 and 0.59), consistent with Q3, where importance partially tracks the cut at the position level while falling short of recovering it.

Table 4: **Downstream utility at equal edit budget.** Injecting the corrupted residual at the same number of positions, the located cut breaks the answer every time; a size-matched random set breaks it 20–53% (1.9–5.0×). Full table in Appendix H.

Family	Pack	n	cut	top-importance	random	cut/random
Qwen2.5-7B	a	40	**1.00**	0.95	0.25	4.0×
Qwen2.5-7B	b	40	**1.00**	1.00	0.20	5.0×
Qwen2.5-14B	a	40	**1.00**	1.00	0.20	5.0×
Qwen2.5-14B	b	40	**1.00**	1.00	0.25	4.0×
Gemma2-9B	a	37	**1.00**	0.86	0.24	4.1×
Gemma2-9B	b	17	**1.00**	0.59	0.53	1.9×

4.5 Q5: External replication on real single-hop, multi-hop, arithmetic, long-context

Table 5 reports the cut on four real question-answering settings; the per-task construction (spans, corruptions, nulls) is in Appendix D.2. In every case the cut is found on a large fraction of examples, stays about one to two positions wide, and the strict control (random sets drawn from clean tokens away from the answer) passes in none of the cells. This holds on single-hop extraction, multi-hop reasoning, arithmetic, and long-context retrieval alike.

Multi-hop reasoning keeps the neck single rather than splitting it. OLMo2's absence on the controlled benchmark reflects a limit of that task's instruction format rather than of the mechanism, since it shows the cut on a real task it solves. Enlarging the long-context candidate pool roughly fivefold rules out a small-pool artifact.

The diagnostic apparatus transfers to real text. The threshold sweep and exhaustive greedy-optimality check from Q1 replicate cleanly on real data for two families spanning normalization schemes: greedy search finds the estimand in 98–100% of checked cases, if anything more reliably than on the controlled benchmark. The importance-matched hard null replicates in sign with a reduced margin ($\approx$3–4× versus $\approx$15× controlled), as expected when real evidence spans are only a few tokens wide (Appendix I.2).

Table 5: **External real-data replication:** single-hop QA (SQuAD), multi-hop QA (HotpotQA), arithmetic (GSM8K), and long-context QA (NQ-long). Enlarged null adds pre-span uncontaminated control tokens; control-only null draws from the pre-span control tokens only. Both nulls use $n_{\text{null}} = 12$ draws per case, so a rate of 0.000 means no draw qualified at that resolution.

Task	Family (norm)	Scale	n	cut-found	mean size	null (enl.)	null (control-only)
SQuAD (single-hop)	Qwen2.5 (pre)	7B	80	0.91	1.42	0.078	**0.000**
SQuAD (single-hop)	Gemma2 (sandwich)	9B	80	0.59	1.38	0.046	**0.000**
HotpotQA (multi-hop)	Qwen2.5 (pre)	7B	60	0.68	1.37	0.059	**0.000**
HotpotQA	OLMo2 (pure-post)	13B	50	0.56	1.75	0.033	**0.000**
GSM8K (arithmetic)	Qwen2.5 (pre)	7B	60	0.60	1.42	0.106	**0.000**
NQ-long	Qwen2.5 (pre)	7B	40	0.88	1.63	0.005	**0.000**
NQ-long	Gemma2 (sandwich)	9B	40	0.75	1.40	0.006	**0.000**

The located positions are stable across how the evidence is corrupted. Repeating the SQuAD search under three corruption methods leaves the control-only null at zero throughout and lowers cut-found rate only modestly on the hardest case (Qwen $0.87 \rightarrow 0.80$; Gemma $0.63 \rightarrow 0.47$), with located positions agreeing 81–92% of the time by Jaccard overlap (Appendix I.3). This robustness is functional as well as positional: the cut itself keeps passing the dual test under a harder corruption (Figure 2C), still passing 100% of Qwen2.5-7B cases and 65% of Gemma2-9B's (Appendix I.4).

4.6 Q6: The cut lives in a band of layers, not only the fixed layer

The main experiments fix the intervention layer L to the middle of the network, raising a fair question: is the bottleneck present across depth, or an artifact of that one choice? We re-run the whole search and control at a grid of depths on Qwen2.5-7B and Gemma2-9B (Table 20). In both families the cut is found on essentially every case through the first half of the network and disappears in the late layers (Figure 1e); this is a band of early-to-middle depths, not one arbitrary layer, consistent with evidence being routed into the answer before the final layers. We report the profile as cut-found rate rather than exact size, since the per-depth sample is small; the band itself is the robust finding. The full per-depth results are in Appendix J (Table 20).

4.7 Coverage summary

The intervention site is validated in seven model families (Table 2), and the cut carries a control-beating separation on real tasks in four of them: Qwen2.5 and Qwen3 (pre-norm), Gemma2 (sandwich-norm), and OLMo2 (post-norm), spanning all three normalization geometries. The null-beating separation also holds on Pythia (GPTNeoX parallel attention), a fifth architecture, adding a further geometry to the coverage (Appendix K.2). Tasks span the controlled benchmark and single-hop, multi-hop, arithmetic, and long-context question answering, over several thousand analyzable cases. The narrow neck is not solely an artifact of instruction tuning (Appendix F.3) and is robust to null-pool composition and size (Appendix F.2). Per-family breakdown in Appendix K.1.

5 Conclusion

We studied a *narrow residual bottleneck*: a small set of evidence-token positions, at a single residual-stream layer, that carries a language model's evidence-to-answer computation on its own, located via a two-sided restore-and-sever test calibrated against a size-matched random control, so its existence is a claim that can fail, not an assertion. Across several families and normalization schemes the cut is narrow, beats this control far more often than random sets of the same size, reappears on real single-hop, multi-hop, arithmetic, and long-context QA, and is a far more effective edit target than a random set at equal budget. Most tellingly, it is not the top of an importance ranking: holding size, test, and model fixed, the highest-importance heads of the same size fail the very test the located cut passes, across four families. The contribution is this object: a positional cut that must pass a two-sided test, calibrated against a size-matched control, and shown to separate from importance under a controlled comparison. To our knowledge, no prior work applies this test, and it gives a falsifiable handle on where evidence is forced to pass.

AI USE STATEMENT

Large language models (GPT and Claude) assisted with LaTeX formatting, prose editing, and debugging of experimental scripts. The authors developed all theoretical results, algorithm designs, and experimental protocols. We reviewed LLM-generated text and rewrote it where needed, and at least two authors tested and verified any LLM-generated code before it was included. The authors take full responsibility for the final content of this work.

ETHICS STATEMENT

This work studies where evidence is routed inside open-weight language models, using only publicly released checkpoints and public question-answering datasets; it involves no human subjects or private data. The located cut is a diagnostic applied after the fact, not a new capability. Its main dual-use risk is that it could inform more targeted edits to a model's answers (Section 4.4), but this requires white-box access that typical deployments do not expose. This risk is not hypothetical: activation-level interventions have already been used to bypass safety behavior directly, by ablating a single direction shown to be necessary and sufficient for refusal (Arditi et al., 2024). Our task, routing evidence to an answer, is not refusal, and locating a cut for one does not locate or weaken the other; but the method generalizes, and the same diagnostic that locates a bottleneck could be pointed at a safety behavior to check whether it is robustly distributed or concentrated in a single fragile point before deployment, an audit rather than an attack. We see its primary value in model auditing, as a falsifiable check of what an answer depends on.

REPRODUCIBILITY STATEMENT

All experiments are reproducible. The analysis code, the routing-family prompt generators, and the per-cell summary and case files are available at the anonymous repository link, whose README maps each table in this paper to the files that produce it. Section 3 specifies the test, the search, and the permutation null. Appendices C and D give the real-data and utility protocols, and Appendix E lists the thresholds and constants. All models are public checkpoints: instruct-tuned Qwen2.5-{0.5,1.5,3,7,14}B (Qwen Team, 2024), Qwen3-{8,14}B (Yang et al., 2025), Llama-3.2-3B (Grattafiori et al., 2024), Mistral-v0.3-7B (Jiang et al., 2023), Gemma2-9B (Gemma Team, 2024), Gemma3-4B, and OLMo2-{7,13}B (Walsh et al., 2024), plus the base models Qwen2.5-7B-Base and Pythia-{1.4,2.8,6.9,12}B (Biderman et al., 2023). Real-data settings use the public SQuAD, HotpotQA, GSM8K, and Natural Questions splits. The controlled benchmark is generated by the released prompt generators.

REFERENCES

Andy Arditi et al. Refusal in language models is mediated by a single direction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Rina Foygel Barber and Emmanuel J. Candès. Controlling the false discovery rate via knockoffs. *Annals of Statistics*, 43(5), 2015.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300, 1995.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning (ICML)*, 2023.

Emmanuel Candès, Yingying Fan, Lucas Janson, and Jinchi Lv. Panning for gold: Model-X knockoffs for high-dimensional controlled variable selection. *Journal of the Royal Statistical Society: Series B*, 80(3), 2018.

Lawrence Chan, Adrià Garriga-Alonso, Nix Goldowsky-Dill, Ryan Greenblatt, Jenny Nitishinskaya, Ansh Radhakrishnan, Buck Shlegeris, and Nate Thomas. Causal scrubbing: a method for rigorously testing interpretability hypotheses. Alignment Forum, 2022.

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

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Gemma Team. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. Localizing model behavior with path patching. *arXiv preprint arXiv:2304.05969*, 2023.

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

Michael Hanna, Sandro Pezzelle, and Yonatan Belinkov. Have faith in faithfulness: Going beyond circuit overlap when faithfully assessing circuits. In *Conference on Language Modeling (COLM)*, 2024.

Stefan Heimersheim and Neel Nanda. How to use and interpret activation patching. *arXiv preprint arXiv:2404.15255*, 2024.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, et al. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 2019.

Jiecheng Lu and Shihao Yang. HyperMLP: An integrated perspective for sequence modeling. *arXiv preprint arXiv:2602.12601*, 2026.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *International Conference on Learning Representations (ICLR)*, 2025.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *International Conference on Learning Representations (ICLR)*, 2023.

Joseph Miller, Bilal Chughtai, and William Saunders. Transformer circuit faithfulness metrics are not robust. In *Conference on Language Modeling (COLM)*, 2024.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *Transformer Circuits Thread*, 2022.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

Aaquib Syed, Can Rager, and Arthur Conmy. Attribution patching outperforms automated circuit discovery. *arXiv preprint arXiv:2310.10348*, 2023.

Curt Tigges, Michael Hanna, Qinan Yu, and Stella Biderman. LLM circuit analyses are consistent across training and scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. Function vectors in large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, et al. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Pete Walsh et al. 2 OLMo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: A circuit for indirect object identification in GPT-2 small. In *International Conference on Learning Representations (ICLR)*, 2023a.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. Label words are anchors: An information flow perspective for understanding in-context learning. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023b.

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

An Yang 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 *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision (ECCV)*, 2014.

A Related work (full discussion)

Circuit discovery localizes behavior to sparse subgraphs. A rich line of work discovers sparse subgraphs carrying a targeted in-context behavior: ACDC (Conmy et al., 2023) prunes edges of a task-specific graph; EAP (Syed et al., 2023) scores edges by gradient approximation; SFC (Marks et al., 2025) finds causal subgraphs of sparse-autoencoder features; Hanna et al. (2024) assess faithfulness by ablating outside a discovered circuit; IOI (Wang et al., 2023a) localizes indirect-object identification under a faithfulness/completeness/minimality triad. These select components by a single importance criterion. We instead search for a bottleneck over token positions, qualified by a two-sided necessity-and-sufficiency test. IOI's "minimality" means "no redundant head"; ours means a small set the evidence must pass through. This is a different object, and it disagrees with importance in practice: the highest-importance heads fail our two-sided test (Section 4.3).

What "importance" means in this literature. Each method above operationalizes importance differently – an edge's ablation effect (ACDC (Conmy et al., 2023), IOI (Wang et al., 2023a)), a gradient approximation of it (EAP (Syed et al., 2023)), a feature's causal effect (SFC (Marks et al., 2025)), an attention-copying pattern confirmed by ablation (retrieval heads (Wu et al., 2025), induction heads (Olsson et al., 2022)), or the effect of patching an averaged per-head vector (function vectors (Todd et al., 2024)) – but each scores one component from one intervention. Section 4.3's baseline, direct logit attribution, is the same kind of measure, held to the same size and test as the located cut so the dissociation is not an artifact of a weak baseline. A bottleneck is not simply a stricter cutoff on these same per-component scores: it is a joint pass/fail outcome on one specific set against a null of equally sized alternatives. A component can top every criterion above and still not be a bottleneck, because none of them asks whether this exact set, not a comparably scored one, is what the evidence is forced through.

Necessity and sufficiency are rarely integrated. Heimersheim & Nanda (2024) establish that denoising and noising patches measure sufficiency and necessity respectively, are "not complements" in general, and make no claim to recover the smallest such set. Miller et al. (2024) show circuit-faithfulness metrics are sensitive to the ablation method. We require the same located set to pass both directions (strictly stronger than either alone) and calibrate it against a size-matched permutation null, making the existence of a bottleneck falsifiable at a fixed intervention operator; sensitivity across operators is orthogonal to this control and discussed in Appendix L. Causal scrubbing (Chan et al., 2022) verifies a hypothesized graph by resampling, without searching for a smallest set or testing dual pass rates against a random baseline.

In-context evidence routing concentrates on a few conduits. Retrieval heads (Wu et al., 2025) identify heads whose ablation degrades long-context retrieval, tested against a random-head control. Function vectors (Todd et al., 2024), label-word anchors (Wang et al., 2023b), and induction heads (Olsson et al., 2022) each localize in-context behavior to a few components. These are importance results at head or position level, in a single setting, under one criterion. Retrieval heads are closest to our random control, but their test is one-sided (ablation only), at head level, and does not ask whether the identified set is a *bottleneck* rather than merely the most important heads. We add the sufficiency half, the position-level cut, and a direct test that separates the cut from an importance ranking.

Scale and circuit size. Circuit-size scaling on Pythia (Tigges et al., 2024) measures component count, not neck width. This measures a different object than ours: a count of graph components, not the width of a positional bottleneck. We use Pythia only as a coverage point (Appendix K.2), not to make a scaling claim of our own. The two results are not in tension: component count and neck width are distinct quantities that need not move together.

Knowledge localization and edit targets. ROME/MEMIT and causal tracing (Meng et al., 2022; 2023) localize stored pretraining facts via corrupted-token restoration (denoising only). Our object localizes the routing of in-context evidence under a dual restore-and-sever test; our downstream-utility result (Table 4) is a targeted-edit demonstration on it.

Neighboring ideas outside the circuit-discovery literature. The corrupt-and-measure logic here predates circuit discovery. Occlusion sensitivity (Zeiler & Fergus, 2014) masks part of an image and reads off the drop in classification probability, the same logic behind our scrubbed counterfactual; unlike our dual test, it corrupts the raw input and never searches for a minimal set at an internal

layer. Causal mediation analysis in language models (Vig et al., 2020) intervenes on neurons and attention heads to trace a behavior's flow through the network, and independently finds it sparse and concentrated, echoing this paper's narrowness claim under a different causal framework.

The knockoff filter (Barber & Candès, 2015; Candès et al., 2018) is related only in philosophy: it controls false discovery by comparing each variable to a synthetic decoy matched to its correlation structure, with a formal FDR guarantee. Our null instead redraws whole position sets from the real candidate pool and estimates chance qualification by direct sampling, a different construction under a shared instinct not to trust a raw score. HyperMLP (Lu & Yang, 2026) offers a complementary view of an attention head as a two-layer MLP over its context, but makes no claim about where evidence must pass. We name these previous works as neighbors, not techniques we adapt.

Positioning summary. Each method surveyed above contributes a piece of this test's machinery, but none combines them: a two-sided pass/fail criterion, calibrated against a size-matched null, on one located set rather than a ranking.

B Notation and definitions summary

- **Minimal causal cut $C^\star$ (estimand)**: the smallest set of source-token residual positions satisfying $\mathrm{restore} \geq 0.8$, $\mathrm{ablate} \geq 0.8$, and both argmax rules.
- **Located cut $\hat{C}$ (estimate)**: the dual-qualifying set returned by greedy search; $|\hat{C}| \geq |C^\star|$, with equality in $\approx 95\%$ of checked cases.
- **Dual cut**: any set (not necessarily smallest) that dual-qualifies.
- **Restore-score / ablate-score**: probability-mass changes of the clean answer under the restore or ablate intervention, normalized by the causal gap (Section 3.3).
- **Causal gap $\Delta(x)$**: the drop in correct-label probability caused by scrubbing; the admissibility criterion (Section 3.1).
- **Permutation null**: random size-matched subset of positions from the relevant-group pool or from all evidence positions; the control-only null draws from pre-span control tokens only. Each per-case rate is estimated from n_{null} draws and resolved to $1/n_{\text{null}}$. Table columns headed "null (rel.)" report this relevant-pool variant.
- **Inter-block residual site**: the tensor entering block $L{+}1$ after all block-L normalizations; validated norm-invariant.

C Extended protocol details

Four routing families on the controlled benchmark. The controlled benchmark contains four routing families (mode-pick, route-pick, tool-mode-pick, claim-verify). Mode-pick, route-pick, and tool-mode-pick share the same binary-selector structure: a selector token picks one of two labeled spans, and the correct answer is a verbatim copy of the selected span. They differ only in surface vocabulary (mode/observation/ registry vs. route/alpha/beta vs. tool/mode/registry). Claim-verify adds a comparison step: it selects a span by the same binary rule, then checks it against a separate claim token and outputs a match/mismatch label instead of a copy. Each family generates prompts with $N = 10$ candidate evidence groups (token spans), a balanced selector, and a single-token answer. The clean-vs-scrubbed contrast flips exactly one branch; the selector is preserved. The generator uses a stable per-case RNG (crc32-seeded) to draw distinct single-token label fillings, giving $\sim$ 58–60 unique configurations per family/rank. Every cell uses ≥ 150 prompts before the validity filter; capped at 60–100 valid cases after the clean-correct$\wedge$scrub-harm filter.

Restore- and ablate-score formulas. The restore-score is the probability-mass increase of the clean answer when the scrubbed residual is replaced at set S by the clean residual, minus the pre-restore mass; the ablate-score is the analogous drop for the ablated direction. Both scores are normalized by the causal gap $\Delta(x)$, so that patching $S = \varnothing$ gives 0 and patching the full evidence region gives 1; they are real-valued and can fall outside $[0, 1]$ when a patch drives the probability past the opposite reference run. A set dual-qualifies if and only if both scores are ≥ 0.8 and the argmax rules hold. Thresholds and the argmax rule are stated in Section 3.3.

Layer choice. All main-text cells use the mid-network layer (`half`), i.e. $L = \lfloor N_{\text{layers}}/2 \rfloor$; this is a site-uniform choice across families. The finer layer band for the per-head contrast (Table 3) is a late-layer band (last quartile of layers): that comparison ranks heads by direct logit attribution to the decision margin, a proxy that is only faithful close to the unembedding, where a head's write is not yet reshaped by many subsequent layers. This does not conflict with Section 4.6, where the position-level neck fades in these same late layers: a head-level cut is still located on a majority of late-layer cases (Section 4.3), consistent with position-level narrowness and head-level dissociation being different objects, each tested at the depth where it is best measured.

Greedy dual-cut search. Initialize $S = \varnothing$. At each step, for every candidate position $p \notin S$, evaluate the dual score $\min(\text{restore}(S \cup \{p\}), \text{ablate}(S \cup \{p\}))$ and add the position that maximizes it; stop when S dual-qualifies, or when no addition improves the dual score by more than a small tolerance (defaults in the released code). The returned set $\hat{C}$ satisfies $|\hat{C}| \geq |C^\star|$ (Section 3.4).

D Real-task construction and evaluation protocol

This section is organized as: real-task and downstream-utility protocol (settings, null-pool correction, and the utility formula), and real-task construction protocol (the per-task construction table).

D.1 Real-task and downstream-utility protocol

Real-task settings. SQuAD uses two shards of 80 dev-set questions each (480 total valid cases across model families). HotpotQA distractor dev uses two shards of $\sim$60 per model family ($\sim$600 cases across five model families). GSM8K test uses two shards of 30–60 per model family. Natural Questions long uses 40 dev cases per model family with pool $\sim$99 candidate positions.

Null pools for real data. A null drawn only from the answer-span positions ($\approx$ the cut) would sit trivially close to the cut and give uninformative separation. We therefore use the enlarged null (span + pre-span uncontaminated control tokens) and the control-only null (pre-span control tokens only). The control-only null is the strongest separator and is 0.000 at $n_{\text{null}} = 12$ draws per case in every one of $\sim$26 real-data cells.

Downstream utility (Table 4). We compare, at equal edit budget $|S|$, the answer-break rate of injecting the corrupted (scrubbed) residual at $S =$ (located cut) versus $S =$ (top-importance) versus $S =$ (random relevant). "Answer-break" is $p_{\text{clean-ans}}(\text{clean}) - p_{\text{clean-ans}}(\text{clean} \to_S \text{scrub}) \geq 0.5$.

D.2 Real-task construction protocol

Each real-data setting is turned into a clean-vs-scrubbed evidence-to-answer pair with a token-aligned corruption, matching the controlled benchmark. In every case the clean prompt has a gold evidence span and a known short answer; the scrubbed prompt replaces the gold span with a length-matched distractor so the answer changes, and the candidate pool is the set of context token positions. The permutation null draws from an enlarged pool (all uncontaminated context positions); the control-only null draws from tokens before the answer span. Table 6 lists the per-task construction.

Table 6: Real-task construction. All use single-token-scored answers, a length-matched span corruption, and the enlarged + control-only nulls.

Task	Dataset	Evidence span	Corruption
Single-hop QA	SQuAD dev	gold answer sentence	length-matched distractor sentence
Multi-hop QA	HotpotQA distractor	the two supporting facts	swap a supporting fact for a distractor
Math	GSM8K test	a number in the problem	replace that number
Long-context QA	Natural Questions	gold short-answer span	length-matched distractor span

Admissibility (clean-correct $\wedge$ scrub-harm $\geq \tau_{\text{harm}}$) is applied per task as in Section 3.1; per-cell valid counts, cut-found rates, and both nulls are in the external-results summaries (Appendix I.1). Totals: 1412 valid real-data cases over five model families run and four task types; four model families reach claim-grade (Appendix K). The control-only null is 0.000 in every cell.

E CONSTANTS AND ARTIFACTS

Two constants are fixed throughout this work: the admissibility threshold $\tau_{\text{harm}} = 0.20$ (Section 3.1) and the dual-qualification threshold $\theta = 0.8$ (Section 3.3). The patch layer is fixed a priori to $L = \lfloor \text{layers}/2 \rfloor$. Per-cell n_{null} values vary by experiment and are reported in each table. Every table in the paper is generated from per-cell summary and case files, which are released in the anonymous repository given in the Reproducibility statement; its README maps each table to the files that produce it. Accuracy, reported as `acc` or `clean-acc` in the tables, is the fraction of unscrubbed prompts the model answers correctly.

F ROBUSTNESS AND STABILITY CHECKS (Q1)

This section is organized as: threshold sensitivity, null-pool composition and size, a base-vs-instruct comparison, a per-case significance test, and an exploratory look across model scale.

F.1 THRESHOLD SENSITIVITY AND ROBUSTNESS CONTROLS

Table 7 gives the full threshold sweep summarized in Section 4.1. Pooled over the controlled-benchmark cells, cut-found rate and mean cut size are stable through the reported 0.8 threshold, so the narrow-neck finding is a property of the mechanism, not of the qualification knob.

Table 7: Threshold sensitivity: pooled cut-found rate (with 95% Wilson CI), mean cut size, and narrow-≤ 2 rate across the dual restore/ablate qualification threshold.

Threshold	cut-found (Wilson 95% CI)	mean size	narrow-≤ 2
0.60	0.851 [0.824, 0.875]	2.55	0.709
0.70	0.851 [0.824, 0.875]	2.55	0.709
0.75	0.851 [0.824, 0.875]	2.55	0.709
0.80 (reported)	0.851 [0.824, 0.875]	2.55	0.709
0.85	0.794 [0.763, 0.821]	2.44	0.739
0.90	0.732 [0.699, 0.763]	2.40	0.753

Importance-matched hard null. On Qwen2.5-7B (both packs, $n_{\text{null}} = 40$/case), size-k null sets drawn from the top-$2k$ positions ranked by single-position causal importance dual-qualify at only 0.055 (pack a) and 0.064 (pack b); the size-matched relevant null is 0.034/0.037. The top-k importance set at the position level dual-qualifies 0.56/0.64 of the time, so position-level importance is a partial predictor of the cut, neither irrelevant nor sufficient.

Greedy optimality. Exhaustively testing all size-$(k-1)$ subsets on cases with $k \leq 3$ finds $\hat{C} = C^{\star}$ in 95.2% ($n = 21$) and 95.8% ($n = 24$) of checked cases, so the located cut is the true minimum in almost all small cases.

F.2 NULL-POOL COMPOSITION AND SIZE

The cut-beats-null separation could in principle depend on how the null pool is constructed. We tabulate its sensitivity to both pool composition and pool size.

On the controlled benchmark across 13 cells and seven families, the relevant-only null ranges 0.000–0.042 and the all-evidence null (which includes distractor spans) ranges 0.000–0.008; composition shifts the null's absolute level modestly, and both stay far below the cut-found rate everywhere.

Mean pool size ranges from ≈ 9 positions (GSM8K) to ≈ 99 (NQ-long), an $11\times$ range. The enlarged null falls as the pool grows (0.13 at the smallest pool, 0.005 at the largest), so enlarging the pool does not inflate the null back toward the cut. The control-only null, the strictest tested anywhere in this work, is 0.000 in all 24 cells regardless of composition or scale.

Table 8: Null-pool sensitivity on real data, sorted by mean pool size. The enlarged null falls as the pool grows; the control-only null is 0.000 at every pool size. All rates use $n_{\text{null}} = 12$ draws per case. Representative cells from 24 total.

Task	Model	n	pool size	null (enl.)	null (control)
GSM8K	Gemma2-9B	30	9.4	0.130	**0.000**
GSM8K	Qwen2.5-7B	60	10.0	0.107	**0.000**
SQuAD	Gemma2-9B	80	12.6	0.046	**0.000**
SQuAD	Qwen2.5-7B	80	13.5	0.078	**0.000**
HotpotQA	OLMo2-13B	50	18.4	0.033	**0.000**
HotpotQA	Qwen2.5-7B	60	18.4	0.059	**0.000**
NQ-long	Qwen2.5-7B	40	98.8	0.005	**0.000**
NQ-long	Gemma2-9B	40	98.8	0.006	**0.000**

F.3 Base versus instruction-tuned models

Every model in the main text is instruction-tuned, raising the question of whether the neck is a fine-tuning artifact. We ran the identical, unmodified pipeline on Qwen2.5-7B-Base and compared against the existing Qwen2.5-7B-Instruct results (same architecture and depth; no layer re-diagnosis).

Table 9: Base vs. instruct on the controlled benchmark. The narrow, null-beating neck appears in the base model at essentially the same magnitude. Null rates use $n_{\text{null}} = 12$ draws per case.

Branch	Pack	n	clean-acc	cut-found	mean size	null (rel.)
Instruct	a	100	0.789	1.000	2.56	0.018
Instruct	b	100	0.876	1.000	2.40	0.018
Base	a	60	0.724	1.000	2.73	0.033
Base	b	60	0.632	0.883	2.62	0.005

The base model is somewhat less capable, as expected without instruction tuning, yet cut sizes are within ≈ 0.2 positions of the instruct model, cut-found rate is 0.88–1.00 against 1.00, and null rates remain an order of magnitude below cut-found throughout. The bottleneck is not solely an artifact of instruction tuning.

F.4 A significance test that accounts for testing many cases at once

Section 3.5 and this appendix's null-pool subsection report how often a random size-matched set passes the same test the located cut passes, pooled across many cases. That is a useful summary, but it does not tell us, for any single case, whether the located cut is a statistically meaningful result on its own. Here we run a proper significance test for each case, then correct for the fact that we are testing 100 cases at once.

For each of 100 admissible cases (Qwen2.5-7B-Instruct, both packs), we draw $n_{\text{null}} = 200$ random size-matched sets from the relevant-only pool and ask how extreme the located cut's score is compared to these random draws. We compute this two ways: a simple version that just checks whether a random draw also clears the $\theta = 0.8$ threshold, and a more sensitive version that instead ranks the located cut's raw score against all 200 random scores directly. We then apply the standard Benjamini–Hochberg correction across all 100 resulting p-values, at the usual $q = 0.05$ level.

Table 10: **Per-case significance of the located cut**, corrected for testing 100 cases at once (Qwen2.5-7B-Instruct, $n_{\text{null}} = 200$ draws per case, relevant-only pool, Benjamini–Hochberg at $q = 0.05$).

	Simple (threshold-based)	Rank-based
Mean per-case p-value	0.026	0.010
Cases significant after correction	97/100 (0.97)	99/100 (0.99)

Even after correcting for running 100 tests at once, 97–99% of individual cases are still statistically significant. This confirms that the separation reported in Section 4.1 is a real, case-by-case effect, not something that only shows up after averaging over many cases.

F.5 Exploratory: neck width across model scale

We fix the main analysis at capable model scales and make no claim about how neck width varies with model size. For completeness we report what we observed on the Qwen2.5 ladder (0.5B, 1.5B, 3B, 7B, 14B), using the same within-prompt paired design as elsewhere: each prompt is matched to itself across two scales and cut size is regressed on $\log_2$(scale) with prompt-clustered robust SE.

The finding. Across the ladder, neck width is largely stable, with a single localized exception at the 1.5B→3B step (Table 11). That step narrows significantly ($\hat{\beta} = -0.93$, $t = -5.24$, 151 matched prompts, 60 narrowing vs. 19 widening), and controlling for prompt difficulty $\Delta(x)$ does not remove it. Every adjacent step (0.5B→1.5B below and 3B→7B, 7B→14B above) is flat and not significant, as is a second family (Qwen3 8B→14B), and the mean cut size is non-monotone across the ladder.

Interpretation. We therefore do not read this as a smooth or regime-wide relationship between scale and neck width; it is a single localized transition whose mechanism we do not characterize here.

Candidate explanations. Two are consistent with what we do observe, without our data distinguishing between them: a genuine representational reorganization at this scale band, echoing the broader scaling literature's observation that some capabilities transition discretely rather than smoothly; or a discontinuity specific to the Qwen2.5 1.5B/3B checkpoint pair (e.g. a training or architectural difference), not to scale per se. The difficulty-controlled result (which strengthens the effect) at least rules out the simplest alternative, that the step merely tracks easier or harder prompts. A proper account across families, with a finer-grained continuous neck measure that could resolve changes below the integer cut-size grid, is left to future work.

Table 11: Exploratory within-prompt paired slopes across the Qwen2.5 ladder, plus one Qwen3 step. Pair counts are narrow : widen : tied. Only the 1.5B→3B step is significant; the rest are flat, and mean cut size is non-monotone across scales.

Step	$\hat{\beta}$/doubling	SE	p	pairs	pairs (n:w:t)
Qwen2.5 0.5B→1.5B	-0.064	0.148	0.669	99	29:37:33
Qwen2.5 1.5B→3B	$\mathbf{-0.927}$	0.177	$< 10^{-5}$	151	60:19:72
Qwen2.5 3B→7B	-0.102	0.125	0.416	144	37:38:69
Qwen2.5 7B→14B	-0.141	0.108	0.190	354	29:13:312
Qwen3 8B→14B	-0.05	–	n.s.	80	–

The difficulty-controlled version of the significant step strengthens ($\hat{\beta} = -0.93 \rightarrow -2.40$ when $\Delta(x)$ is added to the within-prompt regression), because $\Delta(x)$ rises across the step and independently pushes cut size in the opposite direction. We note this for completeness; nothing in the main text depends on it.

G Head-level and dose-response detail (Q3)

This section is organized as: head-level case selection and per-pack results, and the full dose-response curve behind the paper's position-budget comparison.

G.1 Head-level dissociation: case selection and per-pack detail

Case selection. Head-level cases are admitted by the same filter used everywhere else in this work: clean-correct $\wedge$ $\Delta(x) \geq \tau_{\text{harm}}$ (Section 3.1). This filter is position-agnostic and makes no reference to whether a position-level cut exists, so the head-level search runs on every admissible case regardless of what the position-level search found on it. A model may therefore have a low position-level cut-found rate and a high head-level one, as Llama-3.2-3B does: the two searches

operate over different candidate sets (token positions versus attention heads), and the head-level candidate set is both larger and finer-grained.

Search and baselines. The head-level search is a greedy joint restore≥ 0.8 / ablate≥ 0.8 search over individual attention heads in a late-layer band, with two size-matched baselines: the top-importance-head set and a random head-set, each drawn at $n_{\text{null}} = 40$ per case. Importance is defined by direct logit attribution to the decision margin.

Table 12: Head-level dissociation, per model × pack. The random-head null is exactly 0.000 (shown in Table 3) in all eight cells ($n_{\text{null}} = 40$ draws per case).

Model	Family (norm)	Pack	clean-acc	n	head-cut found	mean k	top-imp. qualifies
Qwen2.5-7B	Qwen2.5 (pre)	a	1.00	30	0.567	5.06	0.118
Qwen2.5-7B	Qwen2.5 (pre)	b	1.00	30	0.933	5.04	0.179
Gemma2-9B	Gemma2 (sandwich)	a	1.00	40	0.600	8.75	0.000
Gemma2-9B	Gemma2 (sandwich)	b	0.96	40	0.625	8.56	0.040
Qwen3-8B	Qwen3 (pre, QK-norm)	a	1.00	40	0.900	8.08	0.000
Qwen3-8B	Qwen3 (pre, QK-norm)	b	1.00	40	0.800	7.88	0.031
Llama-3.2-3B	Llama3.2 (pre)	a	0.40	40	0.675	5.56	0.000
Llama-3.2-3B	Llama3.2 (pre)	b	0.41	40	0.550	5.86	0.000

Head-cut width is architecture-dependent: Gemma2 and Qwen3, with wider heads, route through necks of mean $k \approx 8$–9, against $k \approx 5$–6 for Qwen2.5 and Llama-3.2. This is why the narrowness claim is stated at the position level while the dissociation claim is stated at the head level. Llama-3.2-3B is informative despite low task accuracy (0.40): it finds a head-cut 61% of the time with top-importance qualifying at exactly 0.00 in both packs, which argues against a large-capable-model explanation of the dissociation.

G.2 How many positions does the cut actually need?

Section 3.4 describes the greedy search that finds the cut, but the main text only reports where it stops. Here we look at the whole search path: as we allow more and more positions, how quickly does each method reach the point where the answer is restored (sufficiency) or broken (necessity)? We compare three ways of choosing which positions to add first: the greedy order used by our search, an order based on per-position importance, and a random order. All three draw from the same pool of 16 candidate positions, on Qwen2.5-7B-Instruct ($n = 80$, both packs combined; random order uses 20 draws per case).

Table 13 shows, for a few position budgets k, what fraction of cases have already met each condition.

Table 13: **How many positions are needed.** Fraction of cases that already meet the sufficiency or necessity condition once k positions are allowed, by ordering (Qwen2.5-7B-Instruct, $n = 80$; random pools 20 draws per case, $n = 1600$).

Condition	Ordering	k=1	k=2	k=4	k=8	k=12	k=16
Sufficiency (restore≥ 0.8)	Located (greedy)	0.000	0.863	0.938	0.950	0.975	1.000
	Top-importance	0.000	0.438	0.550	0.562	0.688	0.988
	Random	0.000	0.013	0.087	0.530	0.921	0.979
Necessity (ablate≥ 0.8)	Located (greedy)	0.875	0.950	0.950	0.950	0.975	1.000
	Top-importance	0.875	0.950	0.950	0.950	0.988	1.000
	Random	0.110	0.329	0.757	0.993	1.000	1.000

Necessity is easy for everyone: all three orderings clear 0.95 by $k = 8$. The real difference is in sufficiency. Half of all cases already have their answer restored after just 2 located positions. Reaching that same success rate takes 3 positions with the importance ordering, and 8 with a random ordering. This is the same gap discussed in Section 4.3, now shown as a full curve instead of a single number.

H DOWNSTREAM-UTILITY DETAIL (Q4)

Table 14 reproduces the full downstream-utility results summarized in Table 4, one row per (family, pack) cell.

Table 14: Downstream utility at equal edit budget, full table. Injecting the corrupted residual at the same number of positions, the located cut breaks the answer 100% of the time; a size-matched random set breaks it 20%–53% (1.9–$5.0\times$).

Family	Pack	n	located cut	top-importance	random	cut/random
Qwen2.5-7B	a	40	**1.00**	0.95	0.25	4.0×
Qwen2.5-7B	b	40	**1.00**	1.00	0.20	5.0×
Qwen2.5-14B	a	40	**1.00**	1.00	0.20	5.0×
Qwen2.5-14B	b	40	**1.00**	1.00	0.25	4.0×
Gemma2-9B	a	37	**1.00**	0.86	0.24	4.1×
Gemma2-9B	b	17	**1.00**	0.59	0.53	1.9×

I EXTERNAL REAL-DATA REPLICATION DETAIL (Q5)

This section is organized as: the full 26-cell external-replication table, diagnostics re-run on real data, corruption-method robustness, and transfer of the located cut to a harder corruption.

I.1 EXTENDED EXTERNAL-REPLICATION TABLE

The 26-cell real-data table (SQuAD, HotpotQA, GSM8K, NQ-long) with per-shard n_{valid}, cut-found rate, narrow-≤ 2 rate, mean cut size, enlarged-null qualify rate, control-only-null qualify rate, and per-cell n_{null} is reproduced from the released repository `RESULTS_TABLES.md`. The short main-text version (Table 5) preserves the invariants: control-only null $= 0.000$ in every cell, cut-found $\geq$ enlarged-null in every cell, mean cut size $\in [1.25, 1.83]$.

I.2 ROBUSTNESS OF THE DIAGNOSTICS ON REAL DATA

The threshold sweep, importance-matched hard null, and exhaustive greedy-optimality check of Appendix F.1 were re-run on real-data QA (SQuAD single-hop, HotpotQA multi-hop) for two families spanning normalization schemes (Qwen2.5-7B pre-norm, Gemma2-9B sandwich-norm), $n = 80$ valid cases per cell. Threshold sensitivity was computed exactly by tracing a single greedy trajectory per case and reading off the smallest qualifying prefix at each threshold, since the selection order is threshold-independent.

Table 15: Real-data diagnostics at the reported threshold $\theta = 0.8$. `null-hard` is the importance-matched null; `greedy`$= C^\star$ is the exhaustive optimality check on cases with $k \leq 3$. Cells come from a separate diagnostic run (40 valid cases from each of two shards, $n_{\text{null}} = 30$ draws per null, 16 pre-span control tokens), so cut-found and enlarged-null values differ slightly from Table 5; the larger control pool lowers the enlarged null, as in Table 8.

Cell	n	cut-found	mean size	null (enl.)	null (control)	null-hard	greedy$= C^\star$
SQuAD / Qwen2.5-7B	80	0.888	1.38	0.021	**0.000**	0.202	1.000
SQuAD / Gemma2-9B	80	0.575	1.50	0.002	**0.000**	0.221	1.000
HotpotQA / Qwen2.5-7B	80	0.713	1.32	0.004	**0.000**	0.229	0.982
HotpotQA / Gemma2-9B	80	0.588	1.26	0.004	**0.000**	0.241	1.000

Cut-found rate declines smoothly and monotonically as θ tightens from 0.60 to 0.90 in all four cells (e.g. SQuAD/Qwen2.5-7B: $0.913 \to 0.875$), matching the controlled pattern. Greedy optimality transfers more strongly than on the controlled benchmark (98.2–100% vs. $\approx 95\%$), as expected when real evidence spans are short (mean cut size 1.2–1.5), leaving greedy little room to overshoot.

The importance-matched hard null transfers in sign with a reduced margin (≈ 3–$4\times$ vs. $\approx 15\times$ controlled): real evidence spans are only a few tokens wide, so restricting the null pool to the "top-

$2k$ important positions" barely restricts it at all. This is a power limit of this specific diagnostic on short spans, not a weakening of the underlying separation. The control-only null remains 0.000 in all four cells.

I.3 CORRUPTION-METHOD ROBUSTNESS

The causal setup relies on a length-matched semantic distractor as the corruption. To test whether the located cut depends on that choice, we built three parallel corruptions of the same underlying SQuAD item, sharing the passage, question, and gold answer and differing only in substitution method: (1) length-matched semantic distractor (the default), (2) random-token corruption, and (3) an adjacent near-miss, a real word from the same passage closest by character distance to the gold span, the most confusable case. All three share the clean context, so the candidate pool is identical and cut location is directly comparable by Jaccard overlap. All three patch by copying the counterfactual residual; operator variation is out of scope here and discussed in Appendix L.

Table 16: Corruption-method robustness on SQuAD: per-method cut statistics. Control-only null uses $n_{\text{null}} = 12$ draws per case.

Model	Corruption method	n	cut-found	mean size	null (control)
Qwen2.5-7B	Length-matched (default)	194	0.871	1.27	**0.000**
Qwen2.5-7B	Random-token	94	0.851	1.43	**0.000**
Qwen2.5-7B	Adjacent/near-miss	222	0.802	1.51	**0.000**
Gemma2-9B	Length-matched (default)	145	0.628	1.48	**0.000**
Gemma2-9B	Random-token	116	0.560	1.48	**0.000**
Gemma2-9B	Adjacent/near-miss	174	0.471	1.67	**0.000**

Table 17: Pairwise overlap of the located positions across corruption methods (items valid and found under all three).

Model	Method pair	n	Jaccard	Exact-match rate
Qwen2.5-7B	Length-matched vs. random	34	0.922	0.853
Qwen2.5-7B	Length-matched vs. near-miss	34	0.907	0.824
Qwen2.5-7B	Random vs. near-miss	34	0.858	0.735
Gemma2-9B	Length-matched vs. random	31	0.919	0.839
Gemma2-9B	Length-matched vs. near-miss	31	0.809	0.645
Gemma2-9B	Random vs. near-miss	31	0.874	0.710

The control-only null is 0.000 in every row, and the located positions agree across methods 81–92% of the time by Jaccard overlap. This is a stronger statement than "a cut exists under every corruption": the same positions are selected regardless of how the evidence is corrupted. Cut-found rate and overlap decline slightly in the order length-matched $\rightarrow$ random-token $\rightarrow$ near-miss, as expected when the distractor is deliberately the most contextually confusable.

I.4 DOES THE CUT STILL WORK UNDER A DIFFERENT CORRUPTION?

Appendix I.3 shows that the same positions tend to get selected no matter how we corrupt the evidence. Here we ask a more direct question: if we locate the cut under one corruption method, and then test those exact positions under a different corruption on the same example, does the cut still pass? We use the items where a cut can be found under all three corruption methods (length-matched, random-token, and near-miss), locate the cut under the default length-matched corruption, and re-test it under the near-miss corruption, the hardest, most confusable case.

Qwen's cut keeps working almost every time. Gemma's cut works about two thirds of the time, with a real, though modest, drop in score. This matches what Appendix I.3 already shows using position overlap instead of a pass/fail test: near-miss is the hardest corruption to handle, and Gemma is more sensitive to it than Qwen. The difference here is that this test checks whether the cut still works, not just whether the same positions get picked.

Table 18: **Does the located cut transfer to a harder corruption?** Cut located under length-matched corruption, re-tested under near-miss corruption, on items where a cut exists under all three methods. Pass rate is Wilson 95% CI; drop is the average fall in score from the original to the transferred test, with a 95% CI.

Model	n	Still passes [95% CI]	Score drop [95% CI]
Qwen2.5-7B	13	1.00 [0.77, 1.00]	$+0.02$ [-0.03, $+0.07$]
Gemma2-9B	17	0.65 [0.41, 0.83]	$+0.19$ [$+0.07$, $+0.34$]

We can also ask the positional question directly on this same set of items: how often is the exact same set of positions chosen under two different corruption methods, and how does that compare to what two random same-size sets would overlap by chance?

Table 19: **How often is the exact same cut chosen under two different corruption methods?** Exact-match rate with Wilson 95% CI; the last column is the average overlap two random same-size sets would show by chance, for reference. Every pair has at least one non-zero-overlap item ("0 items at Jaccard = 0").

Model	Method pair	n	Exact match [95% CI]	Chance overlap
Qwen2.5-7B	Length-matched vs. random	34	0.85 [0.70, 0.94]	0.57
Qwen2.5-7B	Length-matched vs. near-miss	34	0.82 [0.66, 0.92]	0.58
Qwen2.5-7B	Random vs. near-miss	34	0.74 [0.57, 0.85]	0.57
Gemma2-9B	Length-matched vs. random	31	0.84 [0.67, 0.93]	0.53
Gemma2-9B	Length-matched vs. near-miss	31	0.65 [0.47, 0.79]	0.54
Gemma2-9B	Random vs. near-miss	31	0.71 [0.53, 0.84]	0.56

In every pair, for both models, the same exact positions are chosen far more often than chance overlap alone would explain. This agrees with Appendix I.3's Jaccard-overlap numbers on a slightly different (and stricter, all-three-methods-valid) subset of items, and adds the exact-match view alongside the overlap view already reported there.

J LAYER-BAND DEPTH PROFILE (Q6)

Table 20 gives the full per-depth results summarized in Section 4.6: cut-found rate and mean restore/ablate score at each intervention depth tested, for Qwen2.5-7B (28 layers, both packs pooled) and Gemma2-9B (42 layers, pack a).

Table 20: **Layer-band depth profile.** Cut-found rate and mean restore/ablate score at each tested intervention depth (fraction of network layers). A dash means no case at that depth found a qualifying cut, so no restore/ablate score is defined. In both models the cut is present through roughly the first half of the network and absent in the late layers.

Model	Depth (frac. of layers)	n	cut-found	mean restore / ablate
Qwen2.5-7B	0.250	20	1.000	0.983 / 1.000
(28 layers)	0.393	20	1.000	0.984 / 1.000
	0.536	20	1.000	0.993 / 0.999
	0.714	20	0.100	0.857 / 1.000
	0.857	20	0.000	– / –
Gemma2-9B	0.238	10	1.000	0.989 / 0.999
(42 layers)	0.405	10	1.000	0.967 / 0.998
	0.548	10	0.400	0.871 / 0.999
	0.690	10	0.100	0.801 / 0.863
	0.857	10	0.000	– / –

In both models the cut is found on essentially every case through roughly the first half of the network, then disappears in the late layers; this is a band of depths, not a single arbitrary layer.

K COVERAGE DETAIL: PER-GEOMETRY BREAKDOWN AND PYTHIA

This section is organized as: the per-normalization-geometry breakdown and the Pythia coverage cells.

K.1 COVERAGE SUMMARY

Table 21: Coverage summary per normalization geometry. Real-task claim-grade means a null-beating cut on ≥ 2 shards.

Norm geometry	Claim-grade family	Task types covered
pre-norm	Qwen2.5, Qwen3 (+QK)	controlled, single-hop, multi-hop, math, long-context
sandwich-norm	Gemma2	controlled, single-hop, multi-hop, math, long-context
pure-post-norm	OLMo2	multi-hop (controlled routing limited by capability)
GPTNeoX parallel	Pythia (null-beating only)	controlled benchmark

K.2 PYTHIA: NULL-BEATING SEPARATION ON A FIFTH ARCHITECTURE

Pythia (Biderman et al., 2023) uses GPTNeoX-style parallel attention, a geometry outside the pre-, sandwich-, and post-norm families of the main text. We ran our exact pipeline on Pythia 1.4B–12B, 8 cells (Table 22). Wherever a cut is located it beats the permutation null (0.000 at $n_{\text{null}} = 10$ in every cell), so the null-beating separation extends to this architecture. We report Pythia only for this coverage point and make no scaling claim on it.

Table 22: **Pythia coverage cells.** Pythia 1.4B–12B, 8 cells, our exact protocol. Permutation null 0.000 at $n_{\text{null}} = 10$ in every cell where a cut is located.

Cell	Scale	clean-acc	n_{val}	cut-found	mean size	null (rel.)
pythia-1.4b/a	1.4B	0.342	0	–	–	–
pythia-1.4b/b	1.4B	0.312	6	0.50	2.33	0.000
pythia-2.8b/a	2.8B	0.292	3	0.00	–	–
pythia-2.8b/b	2.8B	0.254	3	1.00	3.33	0.000
pythia-6.9b/a	6.9B	0.250	19	0.11	2.00	0.000
pythia-6.9b/b	6.9B	0.221	13	0.31	2.00	0.000
pythia-12b/a	12B	0.487	29	0.45	3.38	0.000
pythia-12b/b	12B	0.342	12	0.17	3.00	0.000

L DISCUSSIONS AND LIMITATIONS

The ablation operator is held fixed. Miller et al. (2024) show that circuit-faithfulness metrics are sensitive to the ablation operator (resample, mean, or zero). Every intervention here patches by copying the counterfactual residual (resample-style). Our corruption-robustness study (Appendix I.3) varies the input corruption while holding the operator fixed, so it shows the located positions are stable across corruptions, not across operators. Mean- and zero-ablation variants are a natural next step, and would test whether the size-matched null's separation is operator-invariant.

Greedy search and the estimand. The reported cut size $|\hat{C}|$ upper-bounds the estimand $|C^\star|$ (Section 3.4); greedy search can overshoot when the true minimum needs two positions that each contribute negligibly alone. Exhaustive testing of size-$(k-1)$ subsets on cases with $k \leq 3$ finds $\hat{C} = C^\star$ in 95.2% ($n = 21$) / 95.8% ($n = 24$) of checked cases on the controlled benchmark and 98.2–100% on real data (Appendix I.2). The null control still compares like-for-like, since both the located cut and the null are size-k.

Search cost, not enumeration, is what scales. The search in Section 3.4 is greedy, not exhaustive: at each step it scores every remaining candidate once and adds the best, so locating a cut costs on the order of $|P| \cdot k$ dual-score evaluations, not $2^{|P|}$. $|P|$ is set by the task's candidate pool (evidence

positions, or heads in a late-layer band for Section 4.3), not by parameter count, so the search itself does not become combinatorially harder at larger scale. What does grow is the cost per evaluation, since each forward pass is more expensive on a larger model and typically requires multi-GPU inference, and, at the head level specifically, $|P|$ itself grows with depth and width, so a ~ 100B-parameter model would need more evaluations as well as costlier ones. We have not measured this directly; a cheaper joint-test approximation would help here specifically, since it targets evaluations per case rather than the model itself.

Scale dependence is not characterized. We fix the analysis at capable model scales (up to 14B parameters) and make no claim about how neck width varies with size. An exploratory look at one family plus one Qwen3 step (Appendix F.5) finds neck width largely stable with a single localized exception; a proper account across families, with a finer-grained neck measure, is left to future work.

Boundary conditions.

1. The neck is a small set, not guaranteed to be a single token: about two to three positions on the controlled benchmark and one to two on real text.
2. The fraction of cuts with at most two positions depends on the model family (Qwen3-8B sits at cut size ≈ 2.8 on the controlled benchmark) and, at least at one localized step, on scale (Appendix F.5).
3. Small, low-accuracy models find cuts less often, but the null still separates wherever a cut is found: on Pythia, cells with at least six valid cases have cut-found rates of 0.11–0.50, and the null is 0.000 in every cell where a cut is located (Appendix K.2).
4. OLMo2 finds no usable cut on the controlled benchmark. We attribute this to capability and instruction-format limits, not to the mechanism, since it shows the neck on a real task it can solve (Table 5), but we did not isolate the cause.
5. Tasks with no extractable evidence span, such as closed-book recall and code generation, lie outside the setting this method addresses.