

Attr-Kit: Efficient Toolkit for No-Decode Source Attribution

Sai Sundaresan^{*}, Archit Gupta^{*} & Debabrata Mahapatra[†]
Adobe Research, India

Abstract

In-context source attribution, the task of identifying which input passages support an LLM-generated response, is critical for trustworthy grounded generation. A naive solution is to make additional autoregressive LLM calls for citation generation, but the added cost and latency discourage practitioners from incorporating transparency features like attribution. We study efficient **no-decode** alternatives that instead leverage the model’s internal activations. To enable efficient attribution, we introduce Attr-Kit, a modular framework that formalizes the task, unifies several prior approaches, and introduces a new activation signal called **Value Flow** (VF). To study the theoretical potential of activation-based attribution, we formulate a subset selection problem to estimate the maximum achievable attribution accuracy for any signal. The oracle analysis reveals that all activation signals contain sufficient information for near-perfect attribution, and that VF reaches this ceiling with the fewest layers computed. Across multiple datasets and model families, VF consistently outperforms existing activation signals. VF-based attribution incurs zero decode tokens yet achieves 1.4x higher accuracy on average than autoregressive citation from the same model, while making significantly fewer LLM calls. Notably, our no-decode methods with mid-sized models (4–8B) substantially narrow the accuracy gap to frontier models such as GPT-5.4 and Claude Opus 4.6.

1 Introduction

When a language model claims that “the plaintiff filed on March 15th” or “studies show a 30% reduction,” how do we know which part of the input context supports that statement? As models process increasingly long contexts - legal briefs, research papers, internal documentation - users need to verify that outputs are grounded in the provided sources. In-context source attribution is the task of identifying which parts of the input context shaped each part of a model’s response.

Attribution is essential across high-stakes applications. In legal and compliance workflows, answers must be traceable to specific regulatory sources before decisions are made (Thomson Reuters, 2026a;b). Enterprise search systems must guide users to relevant sections of internal documentation (Google Cloud, 2026b; Microsoft, 2026; Adobe, 2025a). In web-grounded question answering, users expect to see which websites support the claims in an answer (Google Cloud, 2026a; Microsoft, 2025). Scientific research tools aggregate findings across papers, requiring clear links back to underlying studies (Elicit, 2026b;a). Across these domains, attribution enables users to assess source credibility, audit evidence, and trust the system’s transparency.

Current attribution approaches fall into two paradigms (Saxena et al., 2025): generation-time citation, where models emit citations as tokens during decoding, and post-hoc citation, where attribution is added after generation. Both face fundamental tradeoffs between accuracy and efficiency. Generation-time methods require extra decode tokens, increasing latency. Post-hoc methods that rely on retrieval-based matching avoid this latency but suffer from lower accuracy. Post-hoc methods that use additional model calls to generate citations

^{*}These authors contributed equally

[†]Project lead and corresponding author - dmahapatra@adobe.com

can achieve better accuracy but reintroduce the latency cost. For high-volume production systems, these tradeoffs compound quickly. Google’s AI Overviews serves over 1.5 billion users monthly; Adobe’s Acrobat AI Assistant reaches more than 650 million monthly active users (Google, 2025; Adobe, 2025b). The challenge is to achieve high source attribution accuracy without adding latency — a combination that existing approaches struggle to deliver simultaneously.

This motivates a different approach: extracting attribution directly from model internals in a training-free manner without additional decoding. Recent work suggests that when a model generates a response, its intermediate activations already encode which input tokens influenced the output (Cohen-Wang et al., 2024a; 2025a; Mahapatra et al., 2025). If this information is recoverable from the forward pass, we can identify supporting sources with minimal added latency. We call such methods *no-decode* attribution. These approaches are valuable in two deployment scenarios: (1) model providers can use them to add post-hoc attribution without additionally fine-tuning for producing citations, keeping the generation mechanism identical with no extra decode tokens. (2) third-party users can apply them post-hoc by running a forward pass on a generated response (from a frontier model) through a smaller model, replacing additional autoregressive calls with a single, efficient attribution step. In both settings, the goal is to match or exceed decode-based accuracy while substantially reducing latency and cost.

In this work, we present Attr-Kit, a unified framework for no-decode source attribution. Our key insight is that multiple activation-based signals can be viewed through a common lens as measures of token influence. Within this framework, we introduce *Value Flow* (VF), a novel attribution signal based on the value-projection stream in attention. We formulate source attribution as a subset-selection problem and demonstrate that activation-based signals contain near-perfect attribution information when properly extracted. Across four models and four datasets, VF achieves the strongest practical performance, matching or exceeding same-model autoregressive citation accuracy while substantially reducing latency, and narrows the gap to frontier model baselines. Our main contributions are:

- We introduce Attr-Kit, a unified framework for no-decode source attribution that encompasses and extends prior activation-based approaches.
- We propose Value Flow (VF), a value-direction attribution signal that emerges as the strongest practical method within this framework.
- We formulate source attribution as a subset-selection problem and show that many activation signals contain near-perfect attribution information, with VF reaching this ceiling earliest.
- Across four models and four datasets, we show that no-decode attribution with Attr-Kit beats same-model autoregressive citation accuracy on average while eliminating decode tokens, and closes the gap to frontier citation systems.

2 Related work

Autoregressive citation generation. A common approach is to generate citations directly in model outputs, as seen in systems like Claude and Deep Research (Anthropic, 2026; OpenAI, 2025). Methods include citation-conditioned prompting (Gao et al., 2023b), retrieval-interleaved decoding (Asai et al., 2024), localized attribution queries (Hirsch et al., 2025), and supervised fine-tuning for sentence-level citations (Zhang et al., 2025; Ye et al., 2024), all within evidence-based generation frameworks (Schreieder et al., 2025). These approaches require autoregressive citation decoding, which increases latency as the number of sources grows.

Post-hoc attribution. An alternative paradigm performs attribution after generation (Saxena et al., 2025). Retrieval-based approaches link statements to sources via similarity matching, adding little latency but often misaligning similarity with evidential support. Methods that invoke additional LM calls - citation prompting or entailment verification - achieve higher accuracy but incur autoregressive cost. Hybrid approaches combine both strategies, includ-

ing retrieval-revision pipelines (Gao et al., 2023a), iterative citation-enhanced generation (Li et al., 2024), and agentic search with MCTS (Li & Ng, 2025).

No-decode attribution from model internals. Mechanistic interpretability shows that attribution-relevant information is encoded in transformer activations (Elhage et al., 2025; Mihaila, 2026; Abnar & Zuidema, 2020), motivating direct extraction methods. ContextCite (Cohen-Wang et al., 2024a) estimates source importance via context ablations, while AT2 (Cohen-Wang et al., 2025a) learns a lightweight surrogate over attention features. Other approaches use internal signals: RCStat (Mahapatra et al., 2025) leverages pre-softmax attention logits, Phukan et al. (2024) use hidden states to detect token-level copying, and value-magnitude signals have been explored for adaptive computation (Tong et al., 2025).

Broader interpretability methods. Model-agnostic methods such as LIME (Ribeiro et al., 2016), SHAP (Lundberg & Lee, 2017), and Integrated Gradients (Sundararajan et al., 2017) rely on perturbations, combinatorial computation, or backward passes. Attention-based work studies whether attention explains predictions (Jain & Wallace, 2019; Wiegrefe & Pinter, 2019), how it propagates across layers (Abnar & Zuidema, 2020), and how it combines with value norms (Kobayashi et al., 2020). Maximum-flow graph formulations (Azarkhalili & Libbrecht, 2025) propagate relevance without backward passes but remain more expensive than our no-decode signals.

3 Attr-Kit: No-Decode Source Attribution

Problem setup. Given S source segments, a query, and G target segments, the goal is to rank source segments by how well they support each target segment using a single forward pass of a model with L layers and H attention heads. Attribution is extracted directly from attention during this pass, without additional generation. For efficiency, computation is restricted to a subset of heads $\mathcal{H}_N \subset \{1, \dots, H\}$ with $|\mathcal{H}_N| = N \ll H$, selected via a profiling step (Section 3.3).

Two phases. Attr-Kit operates in two phases (Figure 1). In the *profiling* phase, a small set of labeled or unlabeled examples flows through four stages – Attribution Signal Extraction, Signal Pooling, Head Scoring, and Head Ranking – to produce the selected head set $\mathcal{H}_N$. In the *inference* phase, a new example undergoes a single forward pass (no decoding), and the same Attribution Signal Extraction and Signal Pooling computations are applied, restricted to $\mathcal{H}_N$; source segments are then ranked by adding the pooled signals over selected heads.

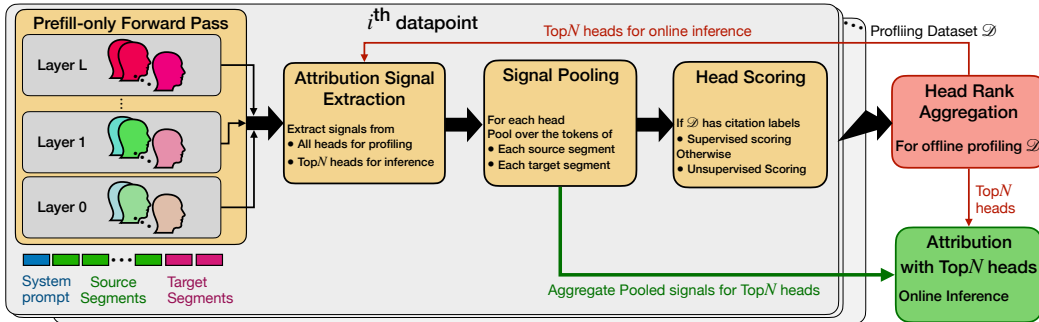


Figure 1: Overview of Attr-Kit. A prefill-only forward pass extracts token-level attribution signals from attention heads, which are pooled into segment-level scores between source and target spans. During profiling, heads are scored and ranked across examples to select a top- N subset. At inference, the same extraction and pooling are applied only to selected heads, and their aggregated signals rank source segments for each target.

3.1 Attribution Signals from Attention Heads

At the most granular level, an attribution signal in a head h measures the influence of a prefix token p on a target token t :

$$\text{signal}_h(t, p) \in \mathbb{R} \quad (1)$$

Let Q_h , K_h , and V_h denote the per-head query, key, and value states, and let A_h denote the causally masked post-softmax attention weights. For the value-based attribution signals, we additionally define the attention output before the output projection,

$$O_h(t) = \sum_p A_h(t, p) V_h(p),$$

and its normalized unit vector

$$\hat{O}_h(t) = O_h(t) / \|O_h(t)\|.$$

We study three types of attribution signals, derived from Q_h , K_h , and V_h , in their raw and normalized form, a total of six different signals; the first two types are from previous literature, and the third, Value Flow, is our contribution. Full definitions for all six signals are given in Appendix A.1.

1. Attention pattern. QK uses the pre-softmax (unnormalized) attention logit, while **Attn** uses the post-softmax (normalized) attention weight. Both rely on the attention pattern alone, without consulting value vectors.

2. Magnitude of Value Vector. MV scales each attention weight by the value norm, $A_h(t, p) \cdot \|V_h(p)\|$, following prior value-calibration work (Tong et al., 2025). High attention is less informative when the value magnitude is small. **MVN** normalizes this by $\|O_h(t)\|$ to account for the overall attention output scale.

3. Value flow (ours). Attention patterns indicate where a token attends, and value norms reflect the scale of retrieved representations. However, neither captures whether the retrieved value contributes to the representation produced by that attention head. A target token may attend strongly to a source token yet produce an attention output that is largely orthogonal to the retrieved value, resulting in limited influence. Conversely, a source value vector highly aligned with the output direction but receiving little attention has no real effect on the target token’s output. We therefore introduce VF, which weights attention by the alignment between a source value vector and the attention output direction:

$$\text{VF: } \text{signal}_h(t, p) = A_h(t, p) (\hat{O}_h(t)^\top V_h(p)) \quad (2)$$

$$\text{VFN: } \text{signal}_h(t, p) = A_h(t, p) (O_h(t)^\top V_h(p)) / \|O_h(t)\|^2 \quad (3)$$

VF captures not just attention or value magnitude, but the contribution of a source token to the target token’s output, making it a more faithful proxy for information flow than either alone. It also retains the same single forward-pass access pattern as other signals in Attr-Kit.

3.2 Signal Pooling

For attribution we need a segment-level score, not a token-level one. Pooling aggregates $\text{signal}_h(t, p)$ over all target tokens $t \in g$ and source tokens $p \in s$ to produce

$$\text{signal}_h^{\text{seg}}(g, s) = \text{pool}(\{\text{signal}_h(t, p) : t \in g, p \in s\}) \in \mathbb{R}, \quad (4)$$

which summarizes how strongly head h connects source segment s to target segment g . This pooled score is used both during head profiling (to score heads) and during inference (to rank source segments). We study five pooling strategies that differ in how they aggregate over target and source tokens; full definitions are in Appendix A.1.

3.3 Profiling

3.3.1 Head Scoring

Head scoring evaluates how well a single head attributes on one profiling example. For example i , the head score is a scalar $\text{score}_h^i \in \mathbb{R}$, computed from the pooled segment scores $\{\text{signal}_h^{\text{seg},i}(g,s)\}_{g,s}$. We study two unsupervised scorers and one supervised scorer.

Supervised: Recall@K (R@K). When attribution labels are available, score_h^i is the recall@K obtained from single-head attribution. For each target segment g , source segments are ranked by $\text{signal}_h^{\text{seg}}(g,s)$ and recall@K is computed against the gold source set, where K equals the number of gold sources. Scores are averaged over target segments:

$$\text{score}_h^i = \frac{1}{G} \sum_g \text{Recall@K}(\text{signal}_h^{\text{seg}}(g, \cdot), \text{gold}_g). \quad (5)$$

Unsupervised: Signal Aggregation When no labels are available, $\text{score}_h^{(i)}$ is the total signal mass directed at source segments. Heads that concentrate signal on source content rather than formatting or query tokens score higher:

$$\text{score}_h^i = \sum_g \sum_s \text{signal}_h^{\text{seg},i}(g,s). \quad (6)$$

Unsupervised: RCStat. RCStat (Mahapatra et al., 2025), from prior literature, operates on pre-softmax QK logits and measures the separation between source-directed and self-directed (target-to-target) attention mass, requiring no labels.

3.3.2 Head Ranking

Given N_{prof} profiling examples, each example i yields a score vector $\mathbf{score}^i = [\text{score}_1^i, \text{score}_2^i, \dots, \text{score}_H^i] \in \mathbb{R}^H$, inducing a ranking over heads. We aggregate these rankings via a rank aggregation operation,

$$\mathbf{r} = \text{rankagg}(\mathbf{score}^1, \dots, \mathbf{score}^{N_{\text{prof}}}), \quad (7)$$

where $\mathbf{r}$ is a permutation of $\{1, \dots, H\}$, and select the top- N heads under $\mathbf{r}$ to form $\mathcal{H}_N$. We study four aggregation strategies: Mean Score, Lower Confidence Bound (LCB), Negative Mean Rank (NMR), and Plackett–Luce (PL) (Plackett, 1975); definitions are in Appendix A.1.

3.4 Inference

At inference time, the attribution score for target segment g and source segment s is obtained by adding the pooled signals of the selected heads:

$$\text{attr}(g,s) = \sum_{h \in \mathcal{H}_N} \text{signal}_h^{\text{seg}}(g,s). \quad (8)$$

Source segments are ranked by $\text{attr}(g,s)$ for each target segment g .

4 Oracle Subset Selection

Practical head selection based on profiling (S3.3, Figure 1) is constrained to use a dataset-level ranking of heads. But this leaves open a more basic question: *how much attribution information is present in a signal at all?* To answer it, we study an oracle that can choose a head subset separately for each example.

Oracle objective. Let $r_{h,s}$ denote the pooled attribution score that head h assigns to source segment s after summing across target segments (i.e., $r_{h,s} = \sum_g \text{signal}_h^{\text{seg}}(g,s)$). Let $x_h \in$

$\{0, 1\}$ indicate whether head h is selected. We restrict the oracle to a subset of heads $\mathcal{H}_l$, comprising all attention heads up to layer l . Given the gold source set $\mathcal{G}$ with $K = |\mathcal{G}|$, the oracle solves the following optimization:

$$\text{Recall@K up to layer } l := \max_{x \in \{0,1\}^{\mathcal{H}_l}} \frac{1}{K} \left| \text{TopK}_s \left(\sum_h x_h r_{h,s} \right) \cap \mathcal{G} \right| \quad (9)$$

where recall@K is as defined in Equation 5. We solve this problem for each example by searching over small head subsets: exhaustive enumeration for up to three heads, with a CP-SAT solver fallback (CPS) (capped at $H_{\max} = 10$ heads, warm-started from the best subset found) when three heads are insufficient to achieve perfect recall.

First-optimal-layer analysis. The oracle also supports a layer-cutoff view. For each example and signal, we find the earliest layer L^* such that the best recall achievable with heads up to layer L^* matches the global optimum. This separates *whether* a signal contains the needed information from *how deep* the model must run before that information becomes available.

Figure 2 summarizes the result. All six signals reach very high oracle recall, confirming that attribution information is broadly present in the head space. The main difference is *how early* it becomes recoverable.

VF achieves 0.996 oracle recall on average and VFN reaches 0.999, while both hit their optimum around layer 3.4. QK also reaches high recall (0.988), but requires roughly twice the depth (mean first-optimal layer 6.6). The challenge, then, is not that activation-based attribution lacks information - it is that practical selection methods must find the heads that already encode it. Full per-signal oracle tables are in Appendix A.8.

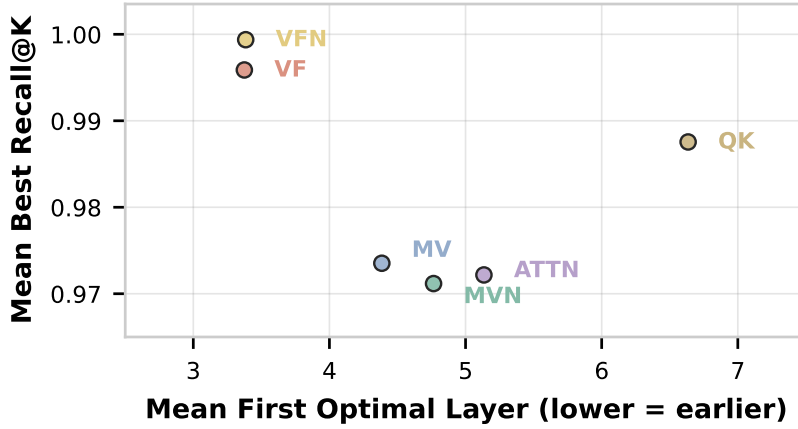


Figure 2: Oracle signal ceiling on Qwen3-8B averaged across four datasets (HotpotQA, MultiRC, VeriGran, GaRAGe - 50 samples each). Each point shows mean best recall versus mean first optimal layer. VF and VFN occupy the top-left region: near-perfect oracle recall with the earliest optimal layers.

5 Experimental Setup

The oracle analysis confirms that citations can be extracted from model activations. Now we evaluate the configurations of Attr-Kit on four datasets and four open-weight models, to provide concrete recommendations for selecting the various components in different scenarios, highlight the latency advantage of our approach and compare against baselines.

Datasets. We evaluate on four datasets (Table 1), each providing source context, query, ground-truth answer, and gold supporting segments. The benchmark suite combines production-style RAG workloads (GaRAGe (Sorodoc et al., 2025) includes passages from SEC filings, AWS DevOps troubleshooting guides, the Enron email corpus, ArXiv abstracts)

with established academic attribution datasets, enabling evaluation spanning compliance, enterprise documentation, scientific literature, internal communications, web content, and multi-hop QA at both sentence and passage granularity. To isolate the performance of the attribution method from variations in answer generation, we attribute the dataset provided ground-truth answer. We use up to 300 samples for profiling and 200 for evaluation.

Table 1: Dataset summary. ATS = avg. target segments per sample; ASS = avg. source segments per sample. [†]VeriGran is smaller: 133 profiling / 146 evaluation samples.

Dataset	Task	Granularity	ATS	ASS
HotpotQA (Yang et al., 2018)	Multi-hop Wikipedia QA	Sentence	1	42
MultiRC (Khashabi et al., 2018)	Multi-sentence RC	Sentence	1	14
VeriGran [†] (Phukan et al., 2024)	Contextual QA	Passage	2.2	64
GaRAGe (Sorodoc et al., 2025)	RAG grounding	Passage	7.8	15

Models. We evaluate four open-weight models spanning two parameter scales 4B and 8B: Qwen3-8B, Llama-3.1-8B-instruct, Phi-4-mini-instruct (4B), and Qwen3-4B-Instruct-2507.

Metrics. We use per-segment recall@ K as the primary metric throughout the main paper. For a target segment g with gold source set Y_g , we rank all source segments by attribution score, set $K = |Y_g|$, and compute the fraction of gold source segments recovered in the top K . Scores are averaged first over target segments within a sample, then across samples. We report results at top-1, top-5, top-10, and top-20 head selections and choose top-10 for baseline comparisons as it is around optimal for most models.

Attr-Kit configuration space. The full sweep covers six attribution signals (QK, Attn, VF, VFN, MV, MVN), 5 signal poolers, 1 supervised scorer (R@K), 2 unsupervised scorers (SAGG, RCStat), and 4 rankers (Mean, LCB, NMR, PL). See Section 3 and Appendix A.1 for definitions.

Ablation settings. Beyond the main baseline comparison, we study two ablations on Llama-3.1-8B at top-10. In the layer-cutoff ablation, we run head selection while restricting candidate heads to layers up to a layer $l \leq L$. In the cross-profiling ablation, we profile on one dataset and evaluate the resulting head rankings on another.

Baselines. We compare against four baseline families:

1. **Same-model autoregressive.** The same open-weight model used in Attr-Kit is prompted to emit citations for each target segment.
2. **Frontier models.** GPT-5.4 and Claude Opus 4.6 with the same per-segment prompt, OpenAI text-embedding-3-Large as a frontier dense retriever, and Claude Opus 4.6 Citations API with native cited generation.
3. **No-decode baselines.** AT2 (Cohen-Wang et al., 2025b), which learns a score estimator over attention features (1 forward pass), and ContextCite (Cohen-Wang et al., 2024b), which uses context ablation for scoring (64 forward passes).
4. **Retrieval baselines.** BM25 lexical retrieval and Qwen3-Embedding-8B dense retrieval. Both are queried with question-plus-answer text and scored via per-segment recall.

Prompting strategy. For autoregressive citation, source segments are tagged and the model is instructed to output only citation tags. We specify to output exactly K citations, where K is the number of gold supporting sources for the target segment, and extract the first K returned tags. Retrieval baselines score each source segment against a query built from the task + target segment. For Claude Citations API, we also label the target segments and instruct the model to reproduce the target verbatim so that citations can be mapped back to the intended segment boundaries. Appendix A.2 provides the prompt templates used.

6 Results

6.1 Comparison across configurations in Attr-Kit

Figure 3 shows the average recall at top-10 heads selected for each signal averaged over the other configuration parameters. Full sweep tables for top-1, top-5, top-10, and top-20 appear in Appendix A.3.

VF signal dominates metrics. In supervised settings VF and VFN achieve 0.72–0.73 average recall. In unsupervised mode, VF is the clear winner. Other value-based signals like MV are competitive but less consistent. QK is the weakest practical choice despite its strong oracle ceiling, suggesting its information is harder to surface with simple global ranking.

Recommended configurations. In supervised settings, use VF or VFN with R@K, SS pooling, and mean aggregation. In unsupervised settings, use VF with SAGG, SS pooling, and rank-consistency aggregation (PL). These defaults capture the dominant pattern in the sweep.

Supervised versus unsupervised. On Llama-3.1-8B, the best unsupervised VF reaches 0.719 versus 0.733 supervised. The gap widens on Qwen, where unsupervised selection is more vulnerable to late-layer heads acting as distractors.

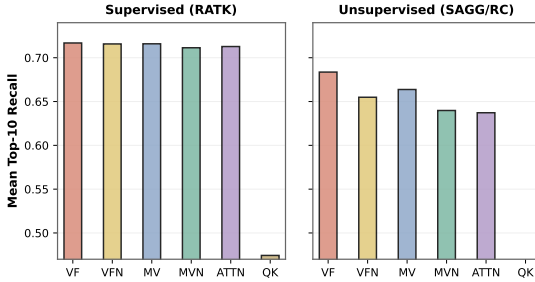


Figure 3: Mean top-10 recall across models for the best configuration of each signal. VF and VFN lead with the lowest variance.

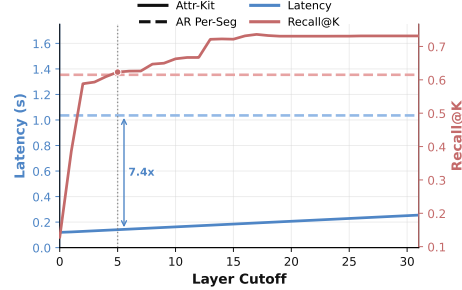


Figure 4: Latency-accuracy tradeoff on Llama-3.1-8B. Attr-Kit is faster at all cut-offs and passes AR Per-Seg recall by layer 5.

6.2 Comparison against baselines

Tables 2 and 3 compare Attr-Kit against the baseline families from Section 5.

Beats same-model autoregressive. Across all four open-weight models, Attr-Kit outperforms per-segment autoregressive citation by 11–25 pp.¹ On Llama-3.1-8B, Attr-Kit reaches 0.740 versus 0.597 for AR Per-Seg. The gap is smallest on HotpotQA and MultiRC, which have single target segments and where AR remains viable, but widens sharply on VeriGran and GaRAGE with longer contexts and multiple target segments.

Faster at inference time. Figure 4 shows that on a timed Llama-3.1-8B benchmark, Attr-Kit is faster at every layer cutoff and exceeds AR recall by layer 5 while being 7.4x faster. Attr-Kit uses 1 call and 0 decode tokens per sample; AR Per-Seg averages 2.35 calls and 19.6 decode tokens per sample.

Beats no-decode baselines. Across the models where AT2 is available, it trails Attr-Kit by 7.5 pp on average. ContextCite is consistently weaker, with Attr-Kit leading it by 16.25 pp on average across the four open-weight models. ContextCite is especially brittle on MultiRC (0.31–0.36 across models), while Attr-Kit maintains 0.68–0.70 recall there. The gap is especially large against dense retrieval on the multi-target datasets VeriGran and GaRAGE, where support must be recovered for multiple related target segments.

¹Parse failures for the AR baseline were negligible ($\leq 1\%$ across all models).

Table 2: Open-weight comparison plus model-agnostic retrieval baselines. Each model block compares Attr-Kit against the same-model autoregressive and available no-decode baselines; the final block reports retrieval baselines. Bold: within 0.5pp of best in each model-specific block.

Model	Method	HPQ	MRC	VGr	GRG	Avg
Qwen3-8B	Attr-Kit	0.625	0.699	0.838	0.750	0.728
	AR Per-Seg	0.651	0.701	0.628	0.488	0.617
	ContextCite	0.404	0.310	0.777	0.702	0.548
Llama-3.1-8B	Attr-Kit	0.648	0.699	0.848	0.766	0.740
	AR Per-Seg	0.637	0.700	0.631	0.422	0.597
	AT2	0.509	0.561	0.841	0.747	0.665
	ContextCite	0.534	0.355	0.770	0.707	0.591
Phi-4-mini	Attr-Kit	0.586	0.698	0.842	0.757	0.721
	AR Per-Seg	0.562	0.645	0.351	0.314	0.468
	AT2	0.497	0.539	0.820	0.731	0.647
	ContextCite	0.496	0.310	0.789	0.719	0.578
Qwen3-4B	Attr-Kit	0.617	0.679	0.846	0.747	0.722
	AR Per-Seg	0.679	0.710	0.569	0.429	0.597
	ContextCite	0.433	0.339	0.720	0.667	0.540
<i>Retrieval baselines</i>						
	BM25	0.598	0.510	0.810	0.657	0.644
	Qwen3-Embedding-8B	0.672	0.483	0.690	0.433	0.570

Outperforms retrieval baselines. Attr-Kit leads BM25 by 9.6 pp on average and Qwen3-Embedding-8B by 16.6 pp on average.

Frontier models. GPT-5.4 and Claude Opus 4.6 in autoregressive mode both reach 0.824 average recall. The best Attr-Kit result (Llama-3.1-8B at 0.733) closes this gap to 9 points, reaching 89% of frontier AR performance. On GaRAGe, Attr-Kit (0.757) is within 1 point of frontier AR (0.763–0.766), and on VeriGran the gap is 4 points.

Among the frontier non-AR baselines, OpenAI Emb.-3-Large reaches 0.606 average recall and Claude Opus 4.6 Citations reaches 0.657, strongest on VeriGran (0.844). Both trail Llama Attr-Kit on HotpotQA, MultiRC, and GaRAGe.

Table 3: Frontier models. Frontier AR baselines reach 0.824 avg recall; best Llama Attr-Kit reaches 0.733. Bold: within 0.5pp of best per column.

Method	HPQ	MRC	VGr	GRG	Avg
<i>Frontier non-AR references</i>					
Attr-Kit (Llama)	0.648	0.699	0.828	0.757	0.733
OpenAI Emb.-3-Large	0.634	0.537	0.723	0.529	0.606
Claude Opus 4.6 Citations	0.665	0.498	0.844	0.621	0.657
<i>Frontier autoregressive</i>					
GPT-5.4 AR Per-Seg	0.879	0.787	0.864	0.766	0.824
Claude Opus 4.6 AR Per-Seg	0.878	0.785	0.871	0.763	0.824

6.3 Ablations

Layer cutoff. Motivated by the oracle finding that attribution information is present in the early layers itself, this ablation restricts head selection to layers up to l , requiring only the first l of L layers to be computed at inference - quantifying the efficiency gains achievable through early-exit beyond eliminating decode tokens. Figure 5 shows that VF forms the

strongest practical frontier: it reaches high recall earlier than alternatives and remains near the envelope as more layers are added.

Cross-profiling stability. This ablation tests how well the heads profiled under each signal on one dataset transfer to another, to see which signals generalize best. Figure 6 shows that unsupervised transfer is dominated by VF (15 of 16 cells), while supervised transfer is more heterogeneous (VFN wins 7 cells, VF 4, ATTN 3, and MV 2).

The appendix contains full sweep tables, oracle tables, and detailed baseline results.

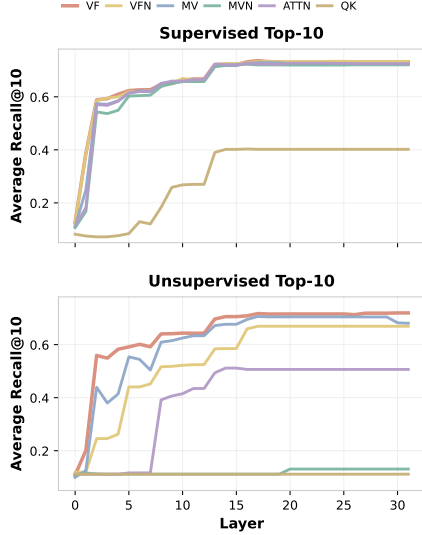


Figure 5: Layer-cutoff curves on Llama-3.1-8B at top-10. VF reaches high recall earliest and traces the strongest frontier.

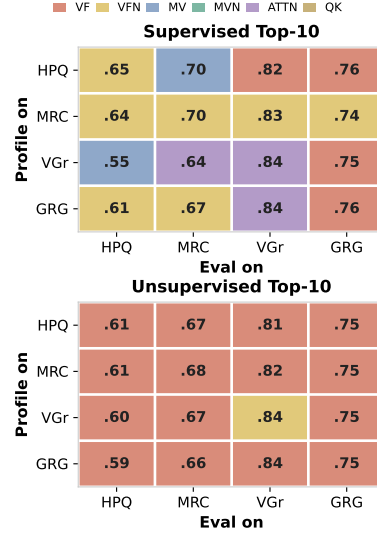


Figure 6: Cross-dataset transfer on Llama-3.1-8B at top-10. Each cell shows the winning signal and absolute recall. Unsupervised transfer is dominated by VF; supervised is more heterogeneous.

7 Conclusion

We introduced Attr-Kit, a modular framework for no-decode source attribution that decomposes the problem into signals, poolers, scorers, and rankers, enabling systematic comparison of prior and new approaches. Our oracle analysis shows that activation signals broadly contain enough information for near-perfect attribution, and that our proposed VF signal reaches this ceiling with the fewest layers. Across datasets and model families, VF also emerges as the strongest practical default. The resulting pipeline avoids citation decoding entirely, beats same-model autoregressive citation on average, and closes the gap to frontier systems.

Limitations. Our profiling selects a static head set; the gap to oracle performance suggests headroom for dynamic head selection. We focus on ranking rather than calibrated selection - turning scores into reliable thresholds for abstention remains open. While our signals are derived from attention, where inter-token interactions occur, rather than the feed-forward sublayers modified by mixture-of-experts routing, generalization to MoE and beyond the 4-8B scale we evaluate is an important direction for future work. Finally, we assume predefined segment boundaries, which is natural for text benchmarks but does not extend directly to modalities such as images where attribution units are not given.

References

- Cp-sat solver — or-tools — google for developers. URL https://developers.google.com/optimization/cp/cp_solver. [Online; accessed 2026-04-01].
- Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4190–4197, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.385. URL <https://aclanthology.org/2020.acl-main.385/>.
- Adobe. Get ai generated answers. Acrobat help documentation, 2025a. URL <https://helpx.adobe.com/acrobat/using/get-ai-generated-answers.html>. Last updated 2025-01-28; accessed 2026-03-29.
- Adobe. Adobe acrobat ai assistant introduces new generative ai features to make understanding contracts easier for everyone. Adobe newsroom, 2025b. URL <https://news.adobe.com/news/2025/02/acrobat-ai-assistant-contracts>. Published 2025-02-04; accessed 2026-03-30.
- Anthropic. Citations. Claude API documentation, 2026. Accessed 2026-03-28.
- Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=hSyW5go0v8>.
- Behrooz Azarkhalili and Maxwell W. Libbrecht. Generalized attention flow: Feature attribution for transformer models via maximum flow. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19954–19974, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.980. URL <https://aclanthology.org/2025.acl-long.980/>.
- Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. Contextcite: attributing model generation to context. In *Proceedings of the 38th International Conference on Neural Information Processing Systems, NIPS '24*, Red Hook, NY, USA, 2024a. Curran Associates Inc. ISBN 9798331314385.
- Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. ContextCite: Attributing model generation to context. *arXiv preprint arXiv:2409.00729*, 2024b. URL <https://arxiv.org/abs/2409.00729>.
- Benjamin Cohen-Wang, Yung-Sung Chuang, and Aleksander Madry. Learning to attribute with attention. *arXiv preprint arXiv:2504.13752*, 2025a.
- Benjamin Cohen-Wang, Yung-Sung Chuang, and Aleksander Madry. Learning to attribute with attention. *arXiv preprint arXiv:2504.13752*, 2025b. URL <https://arxiv.org/abs/2504.13752>.
- Nelson Elhage et al. Circuit tracing: Revealing computational graphs in language models. <https://transformer-circuits.pub/2025/attribution-graphs/methods.html>, 2025.
- Elicit. Elicit reports. Product page, 2026a. URL <https://elicit.com/solutions/reports>. Accessed 2026-03-29.
- Elicit. Ai for scientific research. Product page, 2026b. URL <https://elicit.com/>. Accessed 2026-03-29.
- Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. RARR: Researching and revising what language models say, using language models. In Anna

- Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16477–16508, Toronto, Canada, July 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.910. URL <https://aclanthology.org/2023.acl-long.910/>.
- Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 6465–6488, Singapore, December 2023b. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.398. URL <https://aclanthology.org/2023.emnlp-main.398/>.
- Google. Q1 2025 earnings call: Ceo’s remarks. Google blog, 2025. URL <https://blog.google/inside-google/message-ceo/alphabet-earnings-q1-2025/>. Published 2025-04-24; accessed 2026-03-30.
- Google Cloud. Grounding with google search. Vertex AI documentation, 2026a. URL <https://docs.cloud.google.com/vertex-ai/generative-ai/docs/grounding/grounding-with-google-search>. Accessed 2026-03-29.
- Google Cloud. Grounding with vertex ai search. Vertex AI documentation, 2026b. URL <https://docs.cloud.google.com/vertex-ai/generative-ai/docs/grounding/grounding-with-vertex-ai-search>. Accessed 2026-03-29.
- Eran Hirsch, Aviv Slobodkin, David Wan, Elias Stengel-Eskin, Mohit Bansal, and Ido Dagan. LAQuer: Localized attribution queries in content-grounded generation. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15355–15370, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.746. URL <https://aclanthology.org/2025.acl-long.746/>.
- Sarthak Jain and Byron C. Wallace. Attention is not Explanation. In Jill Burstein, Christy Doran, and Tamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3543–3556, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1357. URL <https://aclanthology.org/N19-1357/>.
- Daniel Khoshnab, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 252–262, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1023. URL <https://aclanthology.org/N18-1023/>.
- Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. Attention is not only a weight: Analyzing transformers with vector norms. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7057–7075, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.574. URL <https://aclanthology.org/2020.emnlp-main.574/>.
- Junyi Li and Hwee Tou Ng. Think&cite: Improving attributed text generation with self-guided tree search and progress reward modeling. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9928–9942, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.490. URL <https://aclanthology.org/2025.acl-long.490/>.

- Weitao Li, Junkai Li, Weizhi Ma, and Yang Liu. Citation-enhanced generation for LLM-based chatbots. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1451–1466, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.79. URL <https://aclanthology.org/2024.acl-long.79/>.
- Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS’17*, pp. 4768–4777, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964. URL <https://dl.acm.org/doi/10.5555/3295222.3295230>.
- Debabrata Mahapatra, Shubham Agarwal, Apoorv Saxena, and Subrata Mitra. Rcstat: A statistical framework for using relative contextualization in transformers. *arXiv preprint arXiv:2506.19549*, 2025.
- Microsoft. Grounding with bing search. Azure AI Foundry documentation, 2025. URL <https://learn.microsoft.com/en-us/azure/ai-foundry/agents/how-to/tools/bing-grounding>. Published 2025-08-07; accessed 2026-03-29.
- Microsoft. Use answer synthesis for citation-backed responses in azure ai search. Microsoft Learn documentation, 2026. URL <https://learn.microsoft.com/en-us/azure/search/agent-retrieval-how-to-answer-synthesis>. Last updated 2026-03-25; accessed 2026-03-29.
- George Mihaila. Learning to explain: Supervised token attribution from transformer attention patterns. *arXiv preprint arXiv:2601.14112*, 2026.
- OpenAI. OpenAI deep research. <https://openai.com/index/introducing-deep-research>, 2025. Accessed: 2026-03-31.
- Anirudh Phukan, Shwetha Somasundaram, Apoorv Saxena, Koustava Goswami, and Balaji Vasan Srinivasan. Peering into the mind of language models: An approach for attribution in contextual question answering. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 11481–11495, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.682. URL <https://aclanthology.org/2024.findings-acl.682/>.
- R. L. Plackett. The analysis of permutations. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 24(2):193–202, 1975. doi: 10.2307/2346567. URL <https://academic.oup.com/jrsssc/article/24/2/193/6953554>.
- Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why Should I Trust You?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’16*, pp. 1135–1144, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939778. URL <https://doi.org/10.1145/2939672.2939778>.
- Yash Saxena, Raviteja Bommireddy, Ankur Padia, and Manas Gaur. Generation-time vs. post-hoc citation: A holistic evaluation of LLM attribution. In *NeurIPS 2025 Workshop on Evaluating the Evolving LLM Lifecycle: Benchmarks, Emergent Abilities, and Scaling*, 2025. URL <https://openreview.net/forum?id=iQyfHmqRJR>.
- Tobias Schreieder, Tim Schopf, and Michael Färber. Attribution, citation, and quotation: A survey of evidence-based text generation with large language models. *arXiv preprint arXiv:2508.15396*, 2025.
- Ionut Teodor Sorodoc, Leonardo F. R. Ribeiro, Rexhina Blloshmi, Christopher Davis, and Adria de Gispert. GaRAGe: A benchmark with grounding annotations for RAG evaluation. *arXiv preprint arXiv:2506.07671*, 2025. URL <https://arxiv.org/abs/2506.07671>.

- Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 3319–3328. JMLR.org, 2017. URL <https://dl.acm.org/doi/10.5555/3305890.3306024>.
- Thomson Reuters. Use ai and cocounsel integration. ONESOURCE Compliance Hub help documentation, 2026a. URL <https://www.thomsonreuters.com/en-us/help/onesource-compliance-hub/get-started/use-ai-and-cocounsel-integration>. Accessed 2026-03-29.
- Thomson Reuters. Use deep research. Westlaw Advantage help documentation, 2026b. URL <https://www.thomsonreuters.com/en-us/help/westlaw-advantage/tools/deep-research>. Accessed 2026-03-29.
- Xin Tong, Zhi Lin, Jingya Wang, and Bo Jin. Have: Head-adaptive gating and value calibration for hallucination mitigation in large language models, 2025. URL <https://arxiv.org/abs/2509.06596>.
- Sarah Wiegrefe and Yuval Pinter. Attention is not not explanation. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 11–20, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1002. URL <https://aclanthology.org/D19-1002/>.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL <https://aclanthology.org/D18-1259/>.
- Xi Ye, Ruoxi Sun, Sercan Arik, and Tomas Pfister. Effective large language model adaptation for improved grounding and citation generation. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6237–6251, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.346. URL <https://aclanthology.org/2024.naacl-long.346/>.
- Jiajie Zhang, Yushi Bai, Xin Lv, Wanjun Gu, Danqing Liu, Minhao Zou, Shulin Cao, Lei Hou, Yuxiao Dong, Ling Feng, and Juanzi Li. LongCite: Enabling LLMs to generate fine-grained citations in long-context QA. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 5098–5122, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.264. URL <https://aclanthology.org/2025.findings-acl.264/>.

A Appendix

A.1 Mathematical Definitions

This section collects the complete formal definitions for all the attribution signals, pooling strategies, and head-ranking strategies studied in Attr-Kit. Head-scoring definitions are given in Section 3.3.1. Throughout, Q_h , K_h , V_h are the per-head query, key, and value states; A_h denotes the causally masked post-softmax attention weights; $O_h(t) = \sum_p A_h(t, p) V_h(p)$ is the attention output before the output projection; and $\hat{O}_h(t) = O_h(t) / \|O_h(t)\|$ is its normalized unit vector.

Attribution Signals from Attention Heads The attribution signal for a head h maps the influence of a prefix position p on a target position t to a scalar $\text{signal}_h(t, p) \in \mathbb{R}$; different formulations are listed in Table 4.

Type	Signal	Definition of $\text{signal}_h(t, p)$
Attention	QK	$Q_h(t)^\top K_h(p) / \sqrt{d}$ (pre-softmax logit)
	Attn	$A_h(t, p)$ (post-softmax weight)
Magnitude of Value Vector	MV	$A_h(t, p) \cdot \ V_h(p)\ $
	MVN	$A_h(t, p) \cdot \ V_h(p)\ / \ O_h(t)\ $
Value Flow (ours)	VF	$A_h(t, p) (\hat{O}_h(t)^\top V_h(p))$
	VFN	$A_h(t, p) (O_h(t)^\top V_h(p)) / \ O_h(t)\ ^2$

Table 4: The six signals studied in Attr-Kit. MV and MVN follow prior value-calibration work (Tong et al., 2025). VF and VFN are our contribution.

Signal pooling. Pooling aggregates token-level signals into segment-level scores, as summarized in Table 5.

$$\text{signal}_h^{\text{seg}}(g, s) = \text{pool}(\{\text{signal}_h(t, p) : t \in g, p \in s\}). \quad (10)$$

The two-letter names indicate aggregation over target then source tokens, where S = sum and M = mean.

Pooler	Definition of $\text{signal}_h^{\text{seg}}(g, s)$
SS	$\sum_{t \in g} \sum_{p \in s} \text{signal}_h(t, p)$ (default; preserves total signal mass)
SM	$\sum_{t \in g} \frac{1}{ s } \sum_{p \in s} \text{signal}_h(t, p)$ (normalizes for source length)
MS	$\frac{1}{ g } \sum_{t \in g} \sum_{p \in s} \text{signal}_h(t, p)$ (normalizes for target length)
MM	$\frac{1}{ g } \sum_{t \in g} \frac{1}{ s } \sum_{p \in s} \text{signal}_h(t, p)$ (normalizes for both)
Max	$\max_{t \in g, p \in s} \text{signal}_h(t, p)$ (peak token-pair signal)

Table 5: The five pooling strategies.

Head ranking. Table 6 lists the head-ranking methods, which aggregate $\text{score}_h^{(i)}$ over the profiling set into hrank_h , and select the top- N heads $\mathcal{H}_N$.

A.2 Prompt templates

Autoregressive citation template. Same-model and frontier AR baselines use the same per-segment prompt structure. All source segments are tagged as [S0], [S1], ..., the answer is shown with the current target segment marked between << and >>, and the model is

Ranker	Definition of hrank_h
Mean	$\frac{1}{N_{\text{prof}}} \sum_i \text{score}_h^{(i)}$
LCB	$\text{mean}_i(\text{score}_h^{(i)}) - \text{std}_i(\text{score}_h^{(i)});$ favors high-scoring, low-variance heads
NMR	$-\text{mean}_i(\text{rank}_h(\{\text{score}_{h'}^{(i)}\}_{h'=1}^H));$ favors heads that consistently rank near the top per example
PL	Plackett–Luce worth estimated from per-example head orderings; favors consistent rank-order advantage

Table 6: The four head-ranking strategies. LCB = Lower Confidence Bound; NMR = Negative Mean Rank; PL = Plackett–Luce.

instructed to output exactly K citations in ranked order, where K is the number of gold supporting segments for that target segment. We use the first K returned tags as the output.

System: You are a source attribution assistant . . . identify exactly K passage(s) that were used to arrive at that segment . . . output only:
 CITATIONS: $S_{<n>}, S_{<n>}, \dots$
 User: Passages: $[S_0] \dots [S_1] \dots$
 Query: {question}
 Answer: {answer upto current segment with <<target segment>> marked}

Retrieval query template. BM25, Qwen3-Embedding-8B, and OpenAI text-embedding-3-Large all score source segments against a query formed from the question + target segment. The lexical and dense retrievers share the same underlying text query, while Qwen3-Embedding prepends its model-card instruction prefix.

Question: {task}
 Answer statement: {target segment}

Qwen3-Embedding-8B only:
 Instruct: Given a question and an answer statement, retrieve passages that directly support the answer statement.
 Query: Question: {task}
 Answer statement: {target text}

Claude Citations API template. For the Claude Citations baseline we label the target segments as well and provide the full target. The model is instructed to reproduce the target verbatim so that citations can be mapped back to the intended segment boundaries.

Question: {task}
 Target segment labels:
 $[T_1] \{ \text{target 1} \}$
 $[T_2] \{ \text{target 2} \}$
 $\dots$
 Full labeled answer: {answer with $[T_1] \dots [/T_1], [T_2] \dots [/T_2], \dots$ }
 Instruction: Copy the labeled answer verbatim, preserve all labels and ordering exactly, and support the copied text with citations.

A.3 Full sweep tables

A.4 Qwen3-8B

Table 7: Qwen3-8B Supervised (RATK): Best 10 configs selecting top-1 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	ratk_ss_mean	SS	0.6040	0.6596	0.8126	0.7489	0.7063
#2	VF	ratk_ms_mean	MS	0.6040	0.6596	0.8126	0.7489	0.7063
#3	MVN	ratk_ss_lcb	SS	0.5965	0.6529	0.8257	0.7394	0.7036
#4	MVN	ratk_ms_lcb	MS	0.5965	0.6529	0.8257	0.7394	0.7036
#5	VF	ratk_ss_lcb	SS	0.5915	0.6596	0.8126	0.7489	0.7031
#6	VF	ratk_ms_lcb	MS	0.5915	0.6596	0.8126	0.7489	0.7031
#7	MV	ratk_ss_mean	SS	0.5882	0.6575	0.8188	0.7446	0.7023
#8	MV	ratk_ss_lcb	SS	0.5882	0.6575	0.8188	0.7446	0.7023
#9	MV	ratk_ms_mean	MS	0.5882	0.6575	0.8188	0.7446	0.7023
#10	MV	ratk_ms_lcb	MS	0.5882	0.6575	0.8188	0.7446	0.7023

Table 8: Qwen3-8B Unsupervised (SAGG + RC): Best 10 configs selecting top-1 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_pl	SS	0.4841	0.4988	0.7937	0.7101	0.6217
#2	VF	sagg_ss_nmr	SS	0.4841	0.4988	0.7937	0.7101	0.6217
#3	VF	sagg_ms_pl	MS	0.4841	0.4988	0.7937	0.7101	0.6217
#4	VF	sagg_ms_nmr	MS	0.4841	0.4988	0.7937	0.7101	0.6217
#5	MV	sagg_ms_lcb	MS	0.4795	0.5421	0.7797	0.6720	0.6183
#6	VF	sagg_ss_mean	SS	0.4787	0.4988	0.7768	0.7101	0.6161
#7	VF	sagg_ms_mean	MS	0.4787	0.4988	0.7768	0.7101	0.6161
#8	MV	sagg_ss_pl	SS	0.4773	0.5421	0.7721	0.6720	0.6159
#9	MV	sagg_ss_nmr	SS	0.4773	0.5421	0.7721	0.6720	0.6159
#10	MV	sagg_ss_mean	SS	0.4773	0.5421	0.7721	0.6720	0.6159

Table 9: Qwen3-8B Supervised (RATK): Best 10 configs selecting top-5 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	MV	ratk_ss_mean	SS	0.6133	0.6987	0.8215	0.7456	0.7198
#2	MV	ratk_ms_mean	MS	0.6133	0.6987	0.8215	0.7456	0.7198
#3	VF	ratk_ss_lcb	SS	0.6250	0.6842	0.8242	0.7408	0.7185
#4	VF	ratk_ms_lcb	MS	0.6250	0.6842	0.8242	0.7408	0.7185
#5	VF	ratk_ss_mean	SS	0.6250	0.6758	0.8243	0.7408	0.7165
#6	VF	ratk_ms_mean	MS	0.6250	0.6758	0.8243	0.7408	0.7165
#7	MVN	ratk_ss_mean	SS	0.6165	0.6896	0.8255	0.7298	0.7153
#8	MVN	ratk_ms_mean	MS	0.6165	0.6896	0.8255	0.7298	0.7153
#9	VFN	ratk_ss_mean	SS	0.6250	0.6754	0.8249	0.7348	0.7150
#10	VFN	ratk_ms_mean	MS	0.6250	0.6754	0.8249	0.7348	0.7150

Table 10: Qwen3-8B Unsupervised (SAGG + RC): Best 10 configs selecting top-5 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ms_lcb	MS	0.5522	0.5350	0.8006	0.7161	0.6510
#2	VF	sagg_ss_nmr	SS	0.5378	0.5350	0.8006	0.7161	0.6474
#3	VF	sagg_ms_nmr	MS	0.5378	0.5350	0.8006	0.7161	0.6474
#4	VF	sagg_ss_pl	SS	0.5112	0.5350	0.8006	0.7161	0.6407
#5	VF	sagg_ms_pl	MS	0.5112	0.5350	0.8006	0.7161	0.6407
#6	VF	sagg_ss_mean	SS	0.5135	0.5225	0.8086	0.7161	0.6402
#7	VF	sagg_ms_mean	MS	0.5095	0.5146	0.8086	0.7161	0.6372
#8	MV	sagg_ss_lcb	SS	0.5436	0.5408	0.7791	0.6702	0.6334
#9	MV	sagg_ss_pl	SS	0.5152	0.5458	0.7867	0.6702	0.6295
#10	MV	sagg_ms_pl	MS	0.5152	0.5458	0.7867	0.6702	0.6295

Table 11: Qwen3-8B Supervised (RATK): Best 10 configs selecting top-10 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	ratk_ss_mean	SS	0.6197	0.6858	0.8232	0.7427	0.7178
#2	VF	ratk_ms_mean	MS	0.6197	0.6858	0.8232	0.7427	0.7178
#3*	VF	ratk_ss_lcb	SS	0.6209	0.6779	0.8123	0.7426	0.7134
#4*	VF	ratk_ms_lcb	MS	0.6209	0.6779	0.8123	0.7426	0.7134
#5*	VFN	ratk_ss_mean	SS	0.6190	0.6717	0.8163	0.7367	0.7109
#6*	VFN	ratk_ms_mean	MS	0.6190	0.6717	0.8163	0.7367	0.7109
#7*	VFN	ratk_ss_lcb	SS	0.6257	0.6567	0.8263	0.7310	0.7099
#8*	VFN	ratk_ms_lcb	MS	0.6257	0.6567	0.8263	0.7310	0.7099
#9*	MVN	ratk_ss_mean	SS	0.6071	0.6829	0.8175	0.7243	0.7079
#10*	MVN	ratk_ms_mean	MS	0.6071	0.6829	0.8175	0.7243	0.7079

Table 12: Qwen3-8B Unsupervised (SAGG + RC): Best 10 configs selecting top-10 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_pl	SS	0.5540	0.5575	0.8040	0.7096	0.6563
#2	VF	sagg_ms_pl	MS	0.5540	0.5575	0.8040	0.7096	0.6563
#3	VF	sagg_ms_lcb	MS	0.5598	0.5442	0.8023	0.7096	0.6540
#4	VF	sagg_ss_nmr	SS	0.5440	0.5575	0.8063	0.6978	0.6514
#5	VF	sagg_ms_nmr	MS	0.5440	0.5575	0.8063	0.6978	0.6514
#6	VF	sagg_ms_mean	MS	0.5272	0.5429	0.8057	0.7123	0.6470
#7	VF	sagg_ss_mean	SS	0.5206	0.5417	0.8057	0.7123	0.6451
#8	VF	sagg_max_pl	MAX	0.5347	0.5192	0.7920	0.7019	0.6369
#9	MV	sagg_ss_lcb	SS	0.5484	0.5450	0.7793	0.6677	0.6351
#10	VF	sagg_max_lcb	MAX	0.5326	0.5133	0.7920	0.7020	0.6350

Table 13: Qwen3-8B Supervised (RATK): Best 10 configs selecting top-20 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1*	VFN	ratk_ss_mean	SS	0.6062	0.6687	0.8249	0.7325	0.7081
#2*	VFN	ratk_ms_mean	MS	0.6062	0.6687	0.8249	0.7325	0.7081
#3*	MV	ratk_ss_mean	SS	0.6079	0.6742	0.8106	0.7280	0.7052
#4*	MV	ratk_ms_mean	MS	0.6079	0.6742	0.8106	0.7280	0.7052
#5*	VF	ratk_ss_mean	SS	0.6152	0.6604	0.8021	0.7426	0.7051
#6*	VF	ratk_ms_mean	MS	0.6152	0.6604	0.8021	0.7426	0.7051
#7*	MVN	ratk_ss_lcb	SS	0.6032	0.6650	0.8249	0.7191	0.7030
#8*	MVN	ratk_ms_lcb	MS	0.6032	0.6650	0.8249	0.7191	0.7030
#9*	ATTN	ratk_ss_mean	SS	0.5790	0.6758	0.8163	0.7401	0.7028
#10*	ATTN	ratk_ms_mean	MS	0.5790	0.6758	0.8163	0.7401	0.7028

Table 14: Qwen3-8B Unsupervised (SAGG + RC): Best 10 configs selecting top-20 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_pl	SS	0.5642	0.5671	0.8088	0.7113	0.6629
#2	VF	sagg_ms_pl	MS	0.5642	0.5671	0.8088	0.7113	0.6629
#3	VF	sagg_ms_lcb	MS	0.5663	0.5558	0.8088	0.7044	0.6588
#4	VF	sagg_ss_lcb	SS	0.5593	0.5608	0.8088	0.7059	0.6587
#5	VF	sagg_ss_nmr	SS	0.5501	0.5550	0.8088	0.7058	0.6549
#6	VF	sagg_ms_nmr	MS	0.5501	0.5550	0.8088	0.7058	0.6549
#7	VF	sagg_ss_mean	SS	0.5355	0.5487	0.8131	0.7092	0.6516
#8	VF	sagg_ms_mean	MS	0.5370	0.5417	0.8128	0.7092	0.6502
#9	VF	sagg_sm_pl	SM	0.5476	0.5542	0.7783	0.7039	0.6460
#10	VF	sagg_mm_pl	MM	0.5476	0.5542	0.7783	0.7039	0.6460

A.5 Llama-3.1-8B

Table 15: Llama-3.1-8B Supervised (RATK): Best 10 configs selecting top-1 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	MVN	ratk_ss_lcb	SS	0.6208	0.6783	0.8303	0.7502	0.7199
#2	MVN	ratk_ms_lcb	MS	0.6208	0.6783	0.8303	0.7502	0.7199
#3	ATTN	ratk_ss_mean	SS	0.6133	0.6887	0.8189	0.7580	0.7197
#4	ATTN	ratk_ms_mean	MS	0.6133	0.6887	0.8189	0.7580	0.7197
#5	ATTN	ratk_ss_lcb	SS	0.6133	0.6887	0.8089	0.7580	0.7172
#6	ATTN	ratk_ms_lcb	MS	0.6133	0.6887	0.8089	0.7580	0.7172
#7	VF	ratk_ss_mean	SS	0.6242	0.6804	0.8038	0.7569	0.7163
#8	VF	ratk_ss_lcb	SS	0.6242	0.6804	0.8038	0.7569	0.7163
#9	VF	ratk_ms_mean	MS	0.6242	0.6804	0.8038	0.7569	0.7163
#10	VF	ratk_ms_lcb	MS	0.6242	0.6804	0.8038	0.7569	0.7163

Table 16: Llama-3.1-8B Unsupervised (SAGG + RC): Best 10 configs selecting top-1 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_pl	SS	0.6071	0.6621	0.8220	0.7569	0.7120
#2	VF	sagg_ss_nmr	SS	0.6071	0.6621	0.8220	0.7569	0.7120
#3	VF	sagg_ms_pl	MS	0.6071	0.6621	0.8220	0.7569	0.7120
#4	VF	sagg_ms_nmr	MS	0.6071	0.6621	0.8220	0.7569	0.7120
#5	VF	sagg_ss_mean	SS	0.5998	0.6621	0.8220	0.7569	0.7102
#6	VF	sagg_ms_mean	MS	0.5998	0.6621	0.8220	0.7569	0.7102
#7	VF	sagg_ms_lcb	MS	0.5998	0.6621	0.8220	0.7569	0.7102
#8	VF	sagg_sm_pl	SM	0.5977	0.6388	0.7875	0.7447	0.6922
#9	VF	sagg_sm_nmr	SM	0.5977	0.6388	0.7875	0.7447	0.6922
#10	VF	sagg_mm_pl	MM	0.5977	0.6388	0.7875	0.7447	0.6922

Table 17: Llama-3.1-8B Supervised (RATK): Best 10 configs selecting top-5 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VFN	ratk_ss_mean	SS	0.6458	0.6825	0.8408	0.7573	0.7316
#2	VFN	ratk_ms_mean	MS	0.6458	0.6825	0.8408	0.7573	0.7316
#3	VF	ratk_ss_mean	SS	0.6458	0.6863	0.8254	0.7656	0.7308
#4	VF	ratk_ms_mean	MS	0.6458	0.6863	0.8254	0.7656	0.7308
#5	VFN	ratk_ss_lcb	SS	0.6396	0.6887	0.8350	0.7530	0.7291
#6	VFN	ratk_ms_lcb	MS	0.6396	0.6887	0.8350	0.7530	0.7291
#7	VF	ratk_ss_lcb	SS	0.6283	0.6875	0.8254	0.7530	0.7236
#8	VF	ratk_ms_lcb	MS	0.6283	0.6875	0.8254	0.7530	0.7236
#9	MV	ratk_ss_mean	SS	0.6397	0.6754	0.8220	0.7548	0.7230
#10	MV	ratk_ms_mean	MS	0.6397	0.6754	0.8220	0.7548	0.7230

Table 18: Llama-3.1-8B Unsupervised (SAGG + RC): Best 10 configs selecting top-5 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_nmr	SS	0.6008	0.6925	0.8482	0.7510	0.7231
#2	VF	sagg_ms_nmr	MS	0.6008	0.6925	0.8482	0.7510	0.7231
#3	VF	sagg_ss_pl	SS	0.5998	0.6742	0.8482	0.7510	0.7183
#4	VF	sagg_ms_pl	MS	0.5998	0.6742	0.8482	0.7510	0.7183
#5	VF	sagg_ms_mean	MS	0.5722	0.6963	0.8482	0.7510	0.7169
#6	VF	sagg_ss_mean	SS	0.5701	0.6787	0.8482	0.7510	0.7120
#7*	VF	sagg_ms_lcb	MS	0.5524	0.6704	0.8482	0.7510	0.7055
#8	VF	sagg_sm_nmr	SM	0.5907	0.6608	0.7942	0.7402	0.6965
#9	VF	sagg_mm_nmr	MM	0.5907	0.6608	0.7942	0.7402	0.6965
#10	VF	sagg_sm_pl	SM	0.5957	0.6554	0.7942	0.7402	0.6964

Table 19: Llama-3.1-8B Supervised (RATK): Best 10 configs selecting top-10 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VFN	ratk_ss_mean	SS	0.6483	0.6987	0.8282	0.7568	0.7330
#2	VFN	ratk_ms_mean	MS	0.6483	0.6987	0.8282	0.7568	0.7330
#3	VFN	ratk_ss_lcb	SS	0.6581	0.6971	0.8282	0.7468	0.7325
#4	VFN	ratk_ms_lcb	MS	0.6581	0.6971	0.8282	0.7468	0.7325
#5	VF	ratk_ss_mean	SS	0.6438	0.6892	0.8316	0.7617	0.7316
#6	VF	ratk_ms_mean	MS	0.6438	0.6892	0.8316	0.7617	0.7316
#7	VF	ratk_ss_lcb	SS	0.6475	0.6937	0.8213	0.7633	0.7315
#8	VF	ratk_ms_lcb	MS	0.6475	0.6937	0.8213	0.7633	0.7315
#9	ATTN	ratk_ss_mean	SS	0.6243	0.6858	0.8350	0.7538	0.7248
#10	ATTN	ratk_ms_mean	MS	0.6243	0.6858	0.8350	0.7538	0.7248

Table 20: Llama-3.1-8B Unsupervised (SAGG + RC): Best 10 configs selecting top-10 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_pl	SS	0.6058	0.6837	0.8397	0.7487	0.7195
#2	VF	sagg_ms_pl	MS	0.6058	0.6837	0.8397	0.7487	0.7195
#3*	VF	sagg_ss_nmr	SS	0.6107	0.6750	0.8397	0.7487	0.7185
#4*	VF	sagg_ms_nmr	MS	0.6107	0.6750	0.8397	0.7487	0.7185
#5*	VF	sagg_ms_lcb	MS	0.5747	0.6721	0.8397	0.7461	0.7081
#6*	VF	sagg_ss_mean	SS	0.5687	0.6767	0.8340	0.7487	0.7070
#7*	VF	sagg_ms_mean	MS	0.5687	0.6742	0.8317	0.7487	0.7058
#8*	VF	sagg_sm_nmr	SM	0.5997	0.6533	0.7803	0.7464	0.6949
#9*	VF	sagg_mm_nmr	MM	0.5997	0.6533	0.7803	0.7464	0.6949
#10*	VF	sagg_sm_pl	SM	0.5857	0.6533	0.7803	0.7464	0.6914

Table 21: Llama-3.1-8B Supervised (RATK): Best 10 configs selecting top-20 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1*	VF	ratk_ss_mean	SS	0.6392	0.6958	0.8314	0.7583	0.7312
#2*	VF	ratk_ms_mean	MS	0.6392	0.6958	0.8314	0.7583	0.7312
#3*	VFN	ratk_ss_lcb	SS	0.6500	0.6825	0.8360	0.7519	0.7301
#4*	VFN	ratk_ms_lcb	MS	0.6500	0.6825	0.8360	0.7519	0.7301
#5*	VF	ratk_ss_lcb	SS	0.6450	0.6817	0.8314	0.7619	0.7300
#6*	VF	ratk_ms_lcb	MS	0.6450	0.6817	0.8314	0.7619	0.7300
#7	MV	ratk_ss_lcb	SS	0.6325	0.6971	0.8291	0.7549	0.7284
#8	MV	ratk_ms_lcb	MS	0.6325	0.6971	0.8291	0.7549	0.7284
#9*	VFN	ratk_ss_mean	SS	0.6438	0.6875	0.8339	0.7482	0.7283
#10*	VFN	ratk_ms_mean	MS	0.6438	0.6875	0.8339	0.7482	0.7283

Table 22: Llama-3.1-8B Unsupervised (SAGG + RC): Best 10 configs selecting top-20 heads.
 Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1*	VF	sagg_ss_pl	SS	0.6080	0.6850	0.8283	0.7519	0.7183
#2*	VF	sagg_ms_pl	MS	0.6080	0.6850	0.8283	0.7519	0.7183
#3*	VF	sagg_ss_nmr	SS	0.6036	0.6642	0.8305	0.7516	0.7125
#4*	VF	sagg_ms_nmr	MS	0.6036	0.6642	0.8305	0.7516	0.7125
#5*	VF	sagg_ms_lcb	MS	0.5876	0.6625	0.8305	0.7529	0.7084
#6*	VF	sagg_ss_mean	SS	0.5799	0.6621	0.8317	0.7513	0.7062
#7*	VF	sagg_ms_mean	MS	0.5834	0.6488	0.8317	0.7528	0.7042
#8	VF	sagg_ss_lcb	SS	0.5897	0.6308	0.8305	0.7510	0.7005
#9*	VF	sagg_sm_pl	SM	0.5932	0.6567	0.7761	0.7466	0.6931
#10*	VF	sagg_mm_pl	MM	0.5932	0.6567	0.7761	0.7466	0.6931

A.6 Phi-4-mini

Table 23: Phi-4-mini Supervised (RATK): Best 10 configs selecting top-1 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	ratk_ss_mean	SS	0.5827	0.6975	0.8186	0.7367	0.7089
#2	VF	ratk_ms_mean	MS	0.5827	0.6975	0.8186	0.7367	0.7089
#3	ATTN	ratk_ss_mean	SS	0.5736	0.6892	0.8268	0.7318	0.7053
#4	ATTN	ratk_ms_mean	MS	0.5736	0.6892	0.8268	0.7318	0.7053
#5	VFN	ratk_ss_mean	SS	0.5801	0.6837	0.8251	0.7308	0.7049
#6	VFN	ratk_ss_lcb	SS	0.5801	0.6837	0.8251	0.7308	0.7049
#7	VFN	ratk_ms_mean	MS	0.5801	0.6837	0.8251	0.7308	0.7049
#8	VFN	ratk_ms_lcb	MS	0.5801	0.6837	0.8251	0.7308	0.7049
#9	ATTN	ratk_ss_lcb	SS	0.5736	0.6892	0.8268	0.7301	0.7049
#10	ATTN	ratk_ms_lcb	MS	0.5736	0.6892	0.8268	0.7301	0.7049

Table 24: Phi-4-mini Unsupervised (SAGG + RC): Best 10 configs selecting top-1 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_pl	SS	0.5180	0.6542	0.8026	0.7367	0.6779
#2	VF	sagg_ss_nmr	SS	0.5180	0.6542	0.8026	0.7367	0.6779
#3	VF	sagg_ms_pl	MS	0.5180	0.6542	0.8026	0.7367	0.6779
#4	VF	sagg_ms_nmr	MS	0.5180	0.6542	0.8026	0.7367	0.6779
#5	VF	sagg_ms_lcb	MS	0.4993	0.6542	0.7926	0.7324	0.6696
#6	VF	sagg_sm_pl	SM	0.5051	0.6467	0.7787	0.7276	0.6645
#7	VF	sagg_sm_nmr	SM	0.5051	0.6467	0.7787	0.7276	0.6645
#8	VF	sagg_mm_pl	MM	0.5051	0.6467	0.7787	0.7276	0.6645
#9	VF	sagg_mm_nmr	MM	0.5051	0.6467	0.7787	0.7276	0.6645
#10	VF	sagg_ss_mean	SS	0.4993	0.5983	0.8186	0.7367	0.6632

Table 25: Phi-4-mini Supervised (RATK): Best 10 configs selecting top-5 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	ratk_ss_lcb	SS	0.5878	0.6746	0.8386	0.7498	0.7127
#2	VF	ratk_ms_lcb	MS	0.5878	0.6746	0.8386	0.7498	0.7127
#3	VF	ratk_ss_mean	SS	0.5753	0.6804	0.8386	0.7565	0.7127
#4	VF	ratk_ms_mean	MS	0.5753	0.6804	0.8386	0.7565	0.7127
#5	ATTN	ratk_ss_mean	SS	0.5782	0.6804	0.8422	0.7472	0.7120
#6	ATTN	ratk_ms_mean	MS	0.5782	0.6804	0.8422	0.7472	0.7120
#7	MV	ratk_ss_mean	SS	0.5851	0.6758	0.8276	0.7476	0.7090
#8	MV	ratk_ms_mean	MS	0.5851	0.6758	0.8276	0.7476	0.7090
#9	VFN	ratk_ss_lcb	SS	0.5837	0.6817	0.8328	0.7377	0.7089
#10	VFN	ratk_ms_lcb	MS	0.5837	0.6817	0.8328	0.7377	0.7089

Table 26: Phi-4-mini Unsupervised (SAGG + RC): Best 10 configs selecting top-5 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ms_mean	MS	0.5412	0.6400	0.8334	0.7472	0.6904
#2	VF	sagg_ss_mean	SS	0.5302	0.6400	0.8334	0.7472	0.6877
#3	VF	sagg_ms_lcb	MS	0.5449	0.6133	0.8306	0.7472	0.6840
#4	VF	sagg_ss_pl	SS	0.5403	0.6208	0.8245	0.7472	0.6832
#5	VF	sagg_ms_pl	MS	0.5403	0.6208	0.8245	0.7472	0.6832
#6	VF	sagg_ss_nmr	SS	0.5387	0.6208	0.8245	0.7472	0.6828
#7	VF	sagg_ms_nmr	MS	0.5387	0.6208	0.8245	0.7472	0.6828
#8	VF	sagg_mm_mean	MM	0.5457	0.6196	0.7840	0.7437	0.6733
#9	MV	sagg_ms_lcb	MS	0.5160	0.6375	0.8105	0.7241	0.6720
#10	MV	sagg_ss_mean	SS	0.5339	0.6029	0.8237	0.7125	0.6683

Table 27: Phi-4-mini Supervised (RATK): Best 10 configs selecting top-10 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1*	VF	ratk_ss_mean	SS	0.5857	0.6787	0.8242	0.7541	0.7107
#2*	VF	ratk_ms_mean	MS	0.5857	0.6787	0.8242	0.7541	0.7107
#3*	VF	ratk_ss_lcb	SS	0.5749	0.6825	0.8303	0.7541	0.7104
#4*	VF	ratk_ms_lcb	MS	0.5749	0.6825	0.8303	0.7541	0.7104
#5	VFN	ratk_ss_lcb	SS	0.5757	0.6800	0.8320	0.7490	0.7092
#6	VFN	ratk_ms_lcb	MS	0.5757	0.6800	0.8320	0.7490	0.7092
#7	ATTN	ratk_ss_lcb	SS	0.5628	0.6879	0.8299	0.7513	0.7080
#8	ATTN	ratk_ms_lcb	MS	0.5628	0.6879	0.8299	0.7513	0.7080
#9*	ATTN	ratk_ss_mean	SS	0.5603	0.6854	0.8285	0.7555	0.7074
#10*	ATTN	ratk_ms_mean	MS	0.5603	0.6854	0.8285	0.7555	0.7074

Table 28: Phi-4-mini Unsupervised (SAGG + RC): Best 10 configs selecting top-10 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_mean	SS	0.5316	0.6554	0.8257	0.7518	0.6911
#2	VF	sagg_ss_nmr	SS	0.5353	0.6404	0.8294	0.7507	0.6890
#3	VF	sagg_ms_nmr	MS	0.5353	0.6404	0.8294	0.7507	0.6890
#4	VF	sagg_ss_pl	SS	0.5316	0.6483	0.8194	0.7499	0.6873
#5	VF	sagg_ms_pl	MS	0.5316	0.6483	0.8194	0.7499	0.6873
#6*	VF	sagg_ms_mean	MS	0.5316	0.6271	0.8257	0.7518	0.6841
#7*	VF	sagg_ms_lcb	MS	0.5278	0.6117	0.8353	0.7535	0.6821
#8	MV	sagg_ss_nmr	SS	0.5202	0.6492	0.8089	0.7275	0.6764
#9	MV	sagg_ms_nmr	MS	0.5202	0.6492	0.8089	0.7275	0.6764
#10	MV	sagg_ms_lcb	MS	0.5218	0.6408	0.8089	0.7211	0.6732

Table 29: Phi-4-mini Supervised (RATK): Best 10 configs selecting top-20 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1*	VF	ratk_ss_lcb	SS	0.5683	0.6729	0.8337	0.7491	0.7060
#2*	VF	ratk_ms_lcb	MS	0.5683	0.6729	0.8337	0.7491	0.7060
#3*	VF	ratk_ss_mean	SS	0.5682	0.6779	0.8257	0.7505	0.7056
#4*	VF	ratk_ms_mean	MS	0.5682	0.6779	0.8257	0.7505	0.7056
#5*	VFN	ratk_ss_mean	SS	0.5628	0.6792	0.8274	0.7477	0.7043
#6*	VFN	ratk_ms_mean	MS	0.5628	0.6792	0.8274	0.7477	0.7043
#7*	MV	ratk_ss_mean	SS	0.5633	0.6675	0.8271	0.7515	0.7024
#8*	MV	ratk_ms_mean	MS	0.5633	0.6675	0.8271	0.7515	0.7024
#9*	VFN	ratk_ss_lcb	SS	0.5645	0.6787	0.8215	0.7425	0.7018
#10*	VFN	ratk_ms_lcb	MS	0.5645	0.6787	0.8215	0.7425	0.7018

Table 30: Phi-4-mini Unsupervised (SAGG + RC): Best 10 configs selecting top-20 heads.
 Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1*	VF	sagg_ss_mean	SS	0.5278	0.6575	0.8191	0.7449	0.6873
#2*	VF	sagg_ms_mean	MS	0.5282	0.6442	0.8260	0.7449	0.6858
#3*	VF	sagg_ss_pl	SS	0.5191	0.6425	0.8260	0.7466	0.6835
#4*	VF	sagg_ms_pl	MS	0.5191	0.6425	0.8260	0.7466	0.6835
#5*	VF	sagg_ss_nmr	SS	0.5262	0.6417	0.8191	0.7447	0.6829
#6*	VF	sagg_ms_nmr	MS	0.5262	0.6417	0.8191	0.7447	0.6829
#7*	VF	sagg_ms_lcb	MS	0.5172	0.6238	0.8260	0.7477	0.6787
#8	MV	sagg_ss_mean	SS	0.5285	0.6350	0.8200	0.7163	0.6750
#9*	MV	sagg_ss_nmr	SS	0.5302	0.6329	0.8200	0.7163	0.6749
#10*	MV	sagg_ms_nmr	MS	0.5302	0.6329	0.8200	0.7163	0.6749

A.7 Qwen3-4B

Table 31: Qwen3-4B Supervised (RATK): Best 10 configs selecting top-1 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	MV	ratk_ss_mean	SS	0.5873	0.6492	0.8125	0.7264	0.6938
#2	MV	ratk_ms_mean	MS	0.5873	0.6492	0.8125	0.7264	0.6938
#3	ATTN	ratk_ss_mean	SS	0.5965	0.6271	0.8061	0.7423	0.6930
#4	ATTN	ratk_ms_mean	MS	0.5965	0.6271	0.8061	0.7423	0.6930
#5	VFN	ratk_ss_mean	SS	0.5875	0.6583	0.8116	0.7101	0.6919
#6	VFN	ratk_ms_mean	MS	0.5875	0.6583	0.8116	0.7101	0.6919
#7	VF	ratk_ss_mean	SS	0.5846	0.6546	0.8142	0.7132	0.6916
#8	VF	ratk_ms_mean	MS	0.5846	0.6546	0.8142	0.7132	0.6916
#9	MV	ratk_ss_lcb	SS	0.5873	0.6492	0.7981	0.7264	0.6902
#10	MV	ratk_ms_lcb	MS	0.5873	0.6492	0.7981	0.7264	0.6902

Table 32: Qwen3-4B Unsupervised (SAGG + RC): Best 10 configs selecting top-1 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_pl	SS	0.5846	0.5300	0.7923	0.7105	0.6543
#2	VF	sagg_ms_pl	MS	0.5846	0.5300	0.7923	0.7105	0.6543
#3	VF	sagg_ms_lcb	MS	0.4910	0.6254	0.7923	0.6825	0.6478
#4	VF	sagg_ss_nmr	SS	0.5487	0.5300	0.7923	0.7105	0.6454
#5	VF	sagg_ms_nmr	MS	0.5487	0.5300	0.7923	0.7105	0.6454
#6	VF	sagg_sm_pl	SM	0.5753	0.5088	0.7721	0.6930	0.6373
#7	VF	sagg_sm_nmr	SM	0.5753	0.5088	0.7721	0.6930	0.6373
#8	VF	sagg_mm_pl	MM	0.5753	0.5088	0.7721	0.6930	0.6373
#9	VF	sagg_mm_nmr	MM	0.5753	0.5088	0.7721	0.6930	0.6373
#10	MV	sagg_ss_pl	SS	0.5117	0.5308	0.7923	0.7038	0.6347

Table 33: Qwen3-4B Supervised (RATK): Best 10 configs selecting top-5 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	ATTN	ratk_ss_mean	SS	0.5998	0.6654	0.8200	0.7467	0.7080
#2	ATTN	ratk_ms_mean	MS	0.5998	0.6654	0.8200	0.7467	0.7080
#3	VFN	ratk_ss_mean	SS	0.5983	0.6646	0.8217	0.7433	0.7070
#4	VFN	ratk_ms_mean	MS	0.5983	0.6646	0.8217	0.7433	0.7070
#5	MV	ratk_ss_mean	SS	0.5935	0.6654	0.8177	0.7417	0.7046
#6	MV	ratk_ms_mean	MS	0.5935	0.6654	0.8177	0.7417	0.7046
#7	MVN	ratk_ss_mean	SS	0.5993	0.6583	0.8251	0.7305	0.7033
#8	MVN	ratk_ms_mean	MS	0.5993	0.6583	0.8251	0.7305	0.7033
#9	MV	ratk_ss_lcb	SS	0.5960	0.6671	0.8143	0.7311	0.7021
#10	MV	ratk_ms_lcb	MS	0.5960	0.6671	0.8143	0.7311	0.7021

A.8 Oracle tables

Table 34: Qwen3-4B Unsupervised (SAGG + RC): Best 10 configs selecting top-5 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_pl	SS	0.5776	0.5671	0.8003	0.7074	0.6631
#2	VF	sagg_ms_pl	MS	0.5776	0.5671	0.8003	0.7074	0.6631
#3	VF	sagg_ss_nmr	SS	0.5763	0.5550	0.8003	0.7074	0.6597
#4	VF	sagg_ms_nmr	MS	0.5763	0.5550	0.8003	0.7074	0.6597
#5	VF	sagg_ms_lcb	MS	0.5346	0.5667	0.8003	0.7074	0.6522
#6	VF	sagg_sm_pl	SM	0.5526	0.5517	0.7723	0.7028	0.6448
#7	VF	sagg_mm_pl	MM	0.5526	0.5517	0.7723	0.7028	0.6448
#8	VF	sagg_sm_nmr	SM	0.5476	0.5550	0.7723	0.7028	0.6444
#9	VF	sagg_mm_nmr	MM	0.5476	0.5550	0.7723	0.7028	0.6444
#10	VF	sagg_mm_lcb	MM	0.5321	0.5671	0.7723	0.7028	0.6436

Table 35: Qwen3-4B Supervised (RATK): Best 10 configs selecting top-10 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	MVN	ratk_ss_mean	SS	0.6146	0.6713	0.8240	0.7350	0.7112
#2	MVN	ratk_ms_mean	MS	0.6146	0.6713	0.8240	0.7350	0.7112
#3	ATTN	ratk_ss_lcb	SS	0.5925	0.6817	0.8228	0.7407	0.7094
#4	ATTN	ratk_ms_lcb	MS	0.5925	0.6817	0.8228	0.7407	0.7094
#5	ATTN	ratk_ss_mean	SS	0.5936	0.6792	0.8186	0.7430	0.7086
#6	ATTN	ratk_ms_mean	MS	0.5936	0.6792	0.8186	0.7430	0.7086
#7	MV	ratk_ss_mean	SS	0.6079	0.6721	0.8129	0.7406	0.7084
#8	MV	ratk_ms_mean	MS	0.6079	0.6721	0.8129	0.7406	0.7084
#9	VFN	ratk_ss_mean	SS	0.6093	0.6604	0.8183	0.7424	0.7076
#10	VFN	ratk_ms_mean	MS	0.6093	0.6604	0.8183	0.7424	0.7076

Table 36: Qwen3-4B Unsupervised (SAGG + RC): Best 10 configs selecting top-10 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ss_pl	SS	0.5923	0.5667	0.8096	0.7012	0.6675
#2	VF	sagg_ms_pl	MS	0.5923	0.5667	0.8096	0.7012	0.6675
#3	VF	sagg_ms_lcb	MS	0.5723	0.5675	0.8074	0.7033	0.6626
#4	VF	sagg_ss_lcb	SS	0.5715	0.5554	0.8142	0.7033	0.6611
#5	VF	sagg_ss_nmr	SS	0.5919	0.5475	0.8028	0.7005	0.6607
#6	VF	sagg_ms_nmr	MS	0.5919	0.5475	0.8028	0.7005	0.6607
#7	MV	sagg_ss_lcb	SS	0.5633	0.5387	0.8091	0.6920	0.6508
#8	MV	sagg_ms_lcb	MS	0.5604	0.5492	0.8009	0.6920	0.6506
#9	VF	sagg_sm_pl	SM	0.5722	0.5567	0.7712	0.6980	0.6495
#10	VF	sagg_mm_pl	MM	0.5722	0.5567	0.7712	0.6980	0.6495

Table 37: Qwen3-4B Supervised (RATK): Best 10 configs selecting top-20 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	MV	ratk_ss_mean	SS	0.6058	0.6721	0.8217	0.7462	0.7115
#2	MV	ratk_ms_mean	MS	0.6058	0.6721	0.8217	0.7462	0.7115
#3	MVN	ratk_ss_lcb	SS	0.6148	0.6750	0.8297	0.7250	0.7111
#4	MVN	ratk_ms_lcb	MS	0.6148	0.6750	0.8297	0.7250	0.7111
#5	VF	ratk_ss_mean	SS	0.6067	0.6746	0.8172	0.7435	0.7105
#6	VF	ratk_ms_mean	MS	0.6067	0.6746	0.8172	0.7435	0.7105
#7*	MVN	ratk_ss_mean	SS	0.6171	0.6654	0.8283	0.7274	0.7096
#8*	MVN	ratk_ms_mean	MS	0.6171	0.6654	0.8283	0.7274	0.7096
#9	MV	ratk_ss_lcb	SS	0.6121	0.6562	0.8195	0.7427	0.7076
#10	MV	ratk_ms_lcb	MS	0.6121	0.6562	0.8195	0.7427	0.7076

Table 38: Qwen3-4B Unsupervised (SAGG + RC): Best 10 configs selecting top-20 heads. Bold = within 0.5pp of best. * = smaller N is better.

#	Signal	Ranker Config	Pool	HPQ	MRC	VGr	GRG	Avg
#1	VF	sagg_ms_lcb	MS	0.5780	0.5850	0.8233	0.7118	0.6745
#2	VF	sagg_ss_pl	SS	0.5827	0.5704	0.8211	0.7076	0.6705
#3	VF	sagg_ms_pl	MS	0.5827	0.5704	0.8211	0.7076	0.6705
#4	VF	sagg_ss_nmr	SS	0.5709	0.5613	0.8220	0.7141	0.6670
#5	VF	sagg_ms_nmr	MS	0.5709	0.5613	0.8220	0.7141	0.6670
#6	MV	sagg_ss_lcb	SS	0.5687	0.5654	0.8182	0.7078	0.6650
#7	MV	sagg_ms_lcb	MS	0.5618	0.5650	0.8182	0.7077	0.6632
#8	VF	sagg_ss_lcb	SS	0.5733	0.5433	0.8251	0.7101	0.6629
#9	MV	sagg_ss_pl	SS	0.5551	0.5558	0.8165	0.7086	0.6590
#10	MV	sagg_ms_pl	MS	0.5551	0.5558	0.8165	0.7086	0.6590

Table 39: Oracle optimal recall@K by signal (Qwen3-8B, 50 examples, ss pooler).

Signal	HPQ	MRC	VGr	GRG	Avg
VF	1.000	1.000	0.990	0.994	0.996
VFN	1.000	1.000	1.000	0.998	0.999
MV	0.980	0.967	0.970	0.977	0.974
MVN	0.975	0.973	0.970	0.966	0.971
ATTN	0.970	0.973	0.970	0.975	0.972
QK	1.000	0.967	1.000	0.984	0.988

Table 40: Mean first optimal layer by signal (Qwen3-8B). Lower = signal reaches optimal recall earlier.

Signal	HPQ	MRC	VGr	GRG	Avg
VF	3.5	2.8	1.4	5.8	3.4
VFN	3.6	3.3	1.5	5.1	3.4
MV	6.1	3.8	1.3	6.3	4.4
MVN	6.2	4.8	1.2	6.9	4.8
ATTN	6.8	5.5	1.3	6.9	5.1
QK	7.0	5.1	6.3	8.1	6.6

Table 41: P95 / worst first optimal layer by signal (Qwen3-8B).

Signal	HPQ	MRC	VGr	GRG
VF	8/9	8/10	2/8	18/33
VFN	8/8	8/19	5/8	12/15
MV	16/31	12/16	3/8	18/29
MVN	20/32	12/27	3/4	18/20
ATTN	16/28	15/23	3/8	16/20
QK	13/16	10/10	12/22	19/24

Table 42: Proven optimal / timeout-best examples (out of 50).

Signal	HPQ	MRC	VGr	GRG
VF	50/0	50/0	49/1	48/2
VFN	50/0	50/0	50/0	49/1
MV	50/0	47/3	50/0	42/8
MVN	49/1	47/3	49/1	40/10
ATTN	50/0	47/3	50/0	41/9
QK	50/0	45/5	50/0	45/5