Let the Model Score Its Own Reasoning: Model-Intrinsic Token Saliency for Chain-of-Thought Compression

Anonymous ACL submission

Abstract

Chain-of-thought (CoT) reasoning improves multi-step problem solving, but long reasoning traces inflate inference cost. Token-level CoT compression reduces this cost by pruning full reasoning chains into shorter traces for model adaptation, making token selection the central challenge. Existing methods often rely on external auxiliary scorers or heuristic signals only indirectly connected to the model's internal answer computation. We therefore shift the focus to a model-intrinsic perspective and propose MIST (Model-Intrinsic Saliency for Token-level CoT compression), which defines a token's importance along two complementary axes: *necessity*, the drop in answer likelihood when a token's internal contribution is removed from the chain, and *sufficiency*, the gain in answer likelihood when that contribution alone is provided. Combining the two yields a unified importance score for pruning. Across four reasoning benchmarks and four models, MIST consistently outperforms baseline methods, suggesting that model-intrinsic saliency provides an effective proxy for reasoning-token importance.

1 Introduction

Chain-of-thought (CoT) reasoning (Wei et al., 2022; Kojima et al., 2022; Wang et al., 2023) has become a standard technique for improving the multi-step problem-solving ability of large language models (LLMs), but produces long intermediate rationales that inflate latency, memory, and serving cost. This has motivated work on CoT compression, which aims to preserve reasoning performance while reducing the cost of reasoning traces. Previous methods obtain compact traces in several ways: prompting or distillation for concise generation (Xu et al., 2025), adaptive reasoning budgets (Han et al., 2025), latent computation that replaces explicit reasoning (Shen et al., 2025), and pruning tokens or steps from a full trace (Xia et al., 2025; Li et al., 2026). We focus on the token pruning setting, where a full CoT trace is pruned to a subset of reasoning tokens that supervise the model to generate shorter chains. This setting makes the central bottleneck explicit: under a retention budget γ, which reasoning tokens should be kept?

Existing token-level compression methods answer this question using signals that are only indirectly tied to the target model's answer computation. TokenSkip-style (Xia et al., 2025) methods, for example, rely on auxiliary scorers (Pan et al., 2024) whose rankings depend on the scorer's own training objective, supervision, and domain assumptions, rather than being derived from the target model's internal reasoning process. While effective, they leave unexplored a more direct and principled path to measuring token importance: deriving it from what the target model itself relies on when forming the answer. We therefore shift the focus from heuristics proxies to a complementary, model-intrinsic perspective. Under this view, a reasoning token is important when the target model's answer computation depends on the internal information carried by that token. Conversely, tokens that make little answer-relevant internal contribution are natural candidates for redundancy under compression. This reframes token-level CoT compression as a model-intrinsic saliency problem: *which reasoning tokens make the most internal contributions to the model's answer computation?*

We formalize this saliency along two complementary axes in the model's residual stream, the internal substrate through which token information propagates across layers. **(Q1) Necessity**: if a token's internal contribution is removed, how much does the ground-truth answer likelihood drop? **(Q2) Sufficiency**: if only that token's internal information is provided, how much of the ground-truth answer likelihood is recovered? Necessity reflects full-chain dependence: it captures tokens whose erasure disrupts the chain's own computation. Suf-

ficiency reflects no-chain recoverability: it captures the contextual hinges whose residuals, given the query, recover answer-relevant information without the rest of the chain. Each view is biased without the other: necessity admits low-information context that is not directly anchored to answer relevance under abundant retention budgets, while sufficiency alone underperforms under aggressive compression, when no single token's residual can substitute for the chain. Combining the two yields a ranking that remains reliable across compression budgets.

Building on this framing, we propose MIST (Model-Intrinsic Saliency for Token-level CoT compression). For each reasoning token, MIST operationalizes the two axes as residual-stream interventions: necessity measures the effect of erasing a token's residual state from the full-chain computation, while sufficiency measures the answer signal recovered by patching that state into a no-chain forward pass. Directly evaluating these interventions requires a separate pass for each token, scaling as $O(T)$ in chain length T which makes token-wise scoring expensive. We show that both axes admit a first-order Taylor linearization in the residual stream, reducing each per-token score to an inner product of gradients and activations, so a single backward pass per axis yields all token scores. MIST then combines the two scores into a unified token-importance score. The highest-scoring tokens are retained to form compressed CoT traces, which are then used to adapt the model to generate shorter reasoning chains.

We evaluate MIST across four reasoning benchmarks, spanning mathematical reasoning (GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021)), and general-domain reasoning (MMLU-Pro (Wang et al., 2024) and BIG-Bench Hard (Suzgun et al., 2023)), and across models, including Qwen2.5-1.5B-Instruct, Qwen2.5-7B-Instruct (Team, 2024), Llama-3.1-8B-Instruct (Grattafiori et al., 2024), and Mistral-7B-Instruct-v0.3 (Jiang et al., 2023a). Extensive results suggest that model-intrinsic saliency serves as a reliable proxy for reasoning-token importance, and that the proposed method consistently improves CoT compression performance across benchmarks and models. In summary, our contributions are:

- We formulate token-level CoT compression as a model-intrinsic saliency problem: rather than asking which tokens appear important to heuristic proxies, we ask which reasoning tokens contribute to the target model's own answer computation, formalized along two complementary axes, necessity and sufficiency.
- We propose MIST, which operationalizes both axes as residual-stream interventions, derives first-order linearizations that reduce per-chain scoring to a single backward pass per axis, and combines the two axes into a unified token-importance score.
- Extensive experimental results across diverse datasets and models demonstrate the effectiveness of our proposed method.

2 Related Work

Efficient chain-of-thought reasoning. Chain-of-thought (CoT) prompting improves multi-step reasoning (Wei et al., 2022; Kojima et al., 2022; Wang et al., 2023) but inflates inference cost, motivating efficient-reasoning methods spanning concise generation (Xu et al., 2025; Munkhbat et al., 2025), adaptive decoding budgets (Han et al., 2025), length-controllable fine-tuning (Ma et al., 2025), latent reasoning (Hao et al., 2024; Shen et al., 2025), and token- or step-level pruning of generated traces (Xia et al., 2025; Li et al., 2026). We focus on the pruning setting.

Token-level CoT pruning. Within the pruning setting, TokenSkip (Xia et al., 2025) constructs compressed CoT supervision with an auxiliary LLMLingua-style token scorer (Jiang et al., 2023b; Pan et al., 2024), rather than the target model's own answer computation. Step-level approaches use signals such as entropy to decide where generation can be skipped (Li et al., 2026). Recent work also studies model-dependent functional importance more directly: GoGI-Skip uses gradient norm of intermediate representations together with dynamic skipping (Zhuang et al., 2025), while likelihood-preserving greedy deletion diagnoses which reasoning tokens can be removed with minimal effect on answer likelihood (Singh and Hakkani-Tür, 2026). In contrast, MIST formulates token importance as model-intrinsic saliency in the target model's residual stream. Rather than relying on an external scorer, a single gradient heuristic, or expensive greedy deletion, MIST measures two complementary intervention-inspired quantities, and combines them into a unified score for token importance.

Model-internal attribution and interventions. Mechanistic interpretability and model-internal attribution methods aim to identify which internal components are important for a model behavior. Causal mediation, causal tracing, and activation patching localize such components by intervening on neurons, heads, or hidden states and measuring changes in model outputs (Vig et al., 2020; Meng et al., 2022; Zhang and Nanda, 2024; Heimersheim and Nanda, 2024; Syed et al., 2024; Ghandeharioun et al., 2024). Residual-stream analyses and direct logit attribution provide a complementary decomposition view, treating Transformer components as writing information into a shared residual stream that contributes to output logits (Elhage et al., 2021; nostalgebraist, 2020). Inspired by this line of work, we define an operational token-saliency measure in the residual stream and use this model-intrinsic signal to quantify the importance of reasoning tokens for CoT compression.

3 MIST: Model-Intrinsic Saliency for Token-level CoT Compression

We present MIST, a model-intrinsic method for token-level CoT compression. Given a query x, a reasoning chain $c = (t_1, \ldots, t_T)$ generated by the target model M, and the answer a, MIST assigns a saliency score to each chain token, retains the top-scoring tokens under a retention budget, and uses the resulting compressed chains to adapt the model for compact reasoning-trace generation. The key idea is to first define what it means for a token to matter to the target model's own answer computation, and then quantify this importance from the model's internal states.

3.1 Problem Formulation

Token-level CoT compression requires a ranking of reasoning tokens. A natural objective is to keep the subset of tokens that best supports the model's answer computation:

$$c^{\star}_{\gamma} = \arg \max_{c' \subseteq c,\ |c'| = \lceil \gamma T \rceil} \log p_M(a \mid x, c'), \qquad (1)$$

where $\gamma \in (0, 1]$ is the retention budget. Solving this objective directly is combinatorial, so practical compression methods reduce the problem to computing per-token importance and retaining the top-ranked subset. In this work, we formulate this importance as *model-intrinsic saliency*: a token is salient if the target model's own answer computation depends on the internal information carried by that token. To make this notion operational, we measure a token's internal contribution through its residual stream, a natural choice since the residual stream is the model's internal computational substrate. We then define this saliency through two complementary axes.

Necessity. Necessity measures what the model loses when the internal contribution of a token is removed from the full reasoning chain. Let h_i denote the residual-stream states associated with token i. We operationalize this removal by zeroing h_i, and then quantify the resulting drop in answer likelihood as the necessity saliency:

$$\phi_i := \log p_M(a \mid x, c) - \log p_M\big(a \mid x, c^{h_i \to 0}\big). \qquad (2)$$

Sufficiency. Sufficiency asks how much of the answer likelihood can be recovered if only token i's internal state is provided, relative to the no chain forward pass. We operationalize this by patching h_i into the no-chain forward pass at the final position and quantify the resulting gain in answer likelihood as the sufficiency saliency:

$$\psi_i := \log p_M\big(a \mid x, \mathrm{patch}(i)\big) - \log p_M(a \mid x, \emptyset). \qquad (3)$$

Why both axes. Necessity and sufficiency capture complementary notions of importance. Necessity scores tokens by full-chain dependence: it identifies tokens whose removal breaks the reasoning computation. Sufficiency scores tokens by no-chain recoverability: a token is sufficient if its contextualized residual raises the answer likelihood from the no-chain baseline. The two axes rank tokens by different criteria, so combining them yields a more reliable ranking than either axis alone.

3.2 Quantifying Model-Intrinsic Saliency in Practice

The formulations above provide conceptually clean measures of token saliency, but their naive computation is prohibitively expensive. Evaluating each token would require an additional forward pass to instantiate either $c^{h_i \to 0}$ or $\mathrm{patch}(i)$, yielding an $O(T)$ cost per chain that is impractical at scale. We instead treat both as residual-stream interventions and linearize $\log p_M$ around the unperturbed forward passes.This yields a practical first-order scorer computable with two backward passes per chain, independent of T.

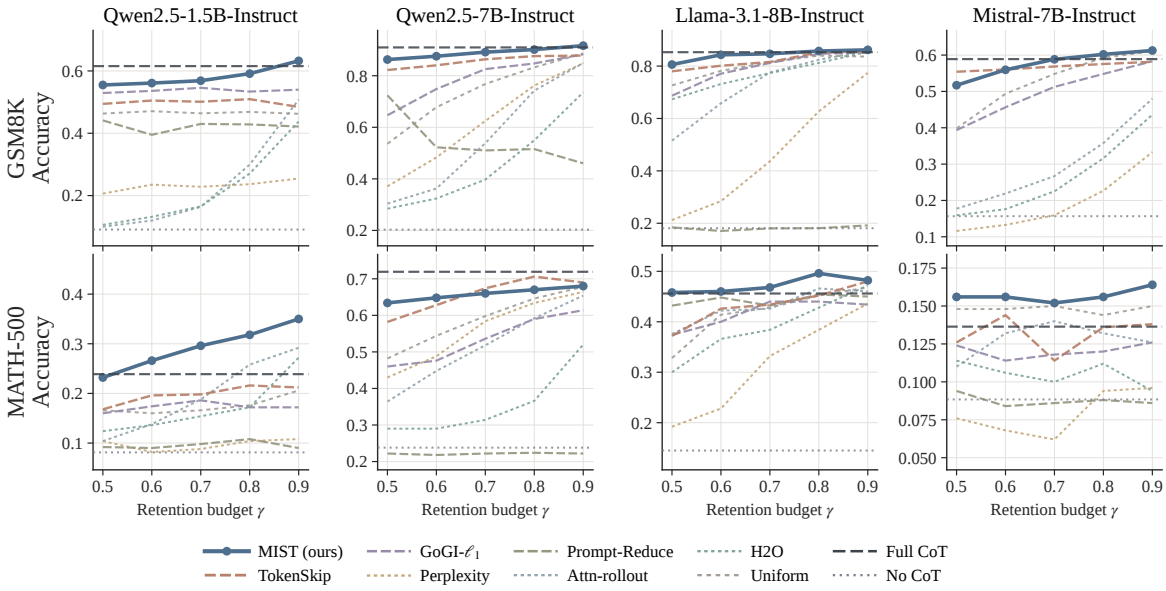

Figure 1: Accuracy across $\gamma \in \{0.5, ..., 0.9\}$ for MIST and all baselines on GSM8K and MATH-500.

Let $h_i^{(l)}$ for $l \in \{1, \ldots, L\}$ denote token i's residual at layer l, with $h_i^{(l),\text{src}}$ and $h_{\text{final}}^{(l),\text{tgt}}$ denoting the unperturbed source and target activations at layer l. The necessity intervention $c^{h_i \to 0}$ perturbs $h_i^{(l),\text{src}}$ along the direction $-h_i^{(l),\text{src}}$; the sufficiency intervention $\text{patch}(i)$ perturbs $h_{\text{final}}^{(l),\text{tgt}}$ along the direction $h_i^{(l),\text{src}} - h_{\text{final}}^{(l),\text{tgt}}$. The answer log-likelihood $\log p_M$ is differentiable in the residual activations, so both saliencies admit a first-order Taylor expansion along these directions.

Necessity term. The first-order expansion of $\log p_M^{\text{src}}$ along $-h_i^{(l),\text{src}}$ identifies the per-layer main term:

$$\widehat{\phi}_i^{(l)} := \langle \nabla_{h_i^{(l)}} \log p_M^{\text{src}},\ h_i^{(l),\text{src}} \rangle, \qquad (4)$$

Both the gradient and the activation in $\widehat{\phi}_i^{(l)}$ are read from the same source forward pass: one backward through $\log p_M^{\text{src}}$ yields $\nabla_{h_i^{(l)}} \log p_M^{\text{src}}$ at every (i, l) pair simultaneously, so the per-chain cost on the necessity axis is one forward and one backward.

Sufficiency term. The first-order expansion of $\log p_M^{\text{tgt}}$ along $h_i^{(l),\text{src}} - h_{\text{final}}^{(l),\text{tgt}}$ identifies the per-layer main term:

$$\widehat{\psi}_i^{(l)} := \langle \nabla_{h_{\text{final}}^{(l)}} \log p_M^{\text{tgt}},\ h_i^{(l),\text{src}} - h_{\text{final}}^{(l),\text{tgt}} \rangle, \quad (5)$$

The gradient is taken at the target's final position; one backward through $\log p_M^{\text{tgt}}$ yields these gradients across all layers l, and the source activations $h_i^{(l),\text{src}}$ are reused from the necessity pass. The full Taylor expansion and remainder bounds for both terms are given in Appendix D.

3.3 The Unified MIST Score

The per-layer terms $\widehat{\phi}_i^{(l)}$ and $\widehat{\psi}_i^{(l)}$ are building blocks; the MIST score combines them into a unified per-token saliency signal by first aggregating across layers along each axis, and then combining the two axes.

Per-axis aggregation. Different layers contribute differently to the answer computation; we therefore adopt a logit-lens-induced (nostalgebraist, 2020) layer weighting: each transformer layer l writes an additive update $h_t^{(l)} - h_t^{(l-1)}$ into the residual-stream state at position t, and the inner product of this update with the unembedding row $W_U[a]$ corresponding to the gold answer token,

$$\bar{c}_l = \frac{1}{T} \sum_{t=1}^{T} \langle h_t^{(l)} - h_t^{(l-1)},\ W_U[a] \rangle, \qquad (6)$$

captures how much that layer's update pushes the chain-averaged hidden state toward the answer; the empirical per-layer distribution of $\bar{c}_l$ across chains is visualized in Fig. 10 (App. G). Aggregating each axis with this shared weight yields the per-token saliencies along the two axes:

$$\widehat{\phi}_i = \sum_l \bar{c}_l \cdot |\widehat{\phi}_i^{(l)}|, \quad \widehat{\psi}_i = \sum_l \bar{c}_l \cdot |\widehat{\psi}_i^{(l)}|. \quad (7)$$

Combining the two axes. Finally, we combine the two axes as[1]:

$$S_i^{\text{MIST}} = \alpha \cdot \widehat{\phi}_i + (1 - \alpha) \cdot \widehat{\psi}_i, \quad \alpha \in [0, 1]. \quad (8)$$

[1] $\widehat{\phi}_i$ and $\widehat{\psi}_i$ are normalized before combining; see App. H for details.

Dataset	Model	Method	Δ Acc. (%)	Comp. rate
GSM8K	Qwen2.5-7B-Inst.	⋆ MIST	↓ 2.4	21.3%
		tokenskip	↓ 4.3	20.0%
		gogi_l1	↓ 11.9	8.9%
		perplexity	↓ 29.2	17.6%
		attn_rollout	↓ 35.1	21.4%
		h2o	↓ 45.2	22.0%
	Mistral-7B-Inst.	⋆ MIST	↓ 1.3	13.4%
		tokenskip	↓ 2.1	10.4%
		gogi_l1	↓ 9.2	14.8%
		perplexity	↓ 39.5	25.6%
		attn_rollout	↓ 28.8	22.5%
		h2o	↓ 32.6	23.0%
MATH-500	Qwen2.5-1.5B-Inst.	⋆ MIST	↑ 5.3	9.3%
		tokenskip	↓ 4.1	10.2%
		gogi_l1	↓ 6.6	3.9%
		perplexity	↓ 14.1	8.5%
		attn_rollout	↓ 4.2	0.0%
		h2o	↓ 6.7	4.7%
	Llama-3.1-8B-Inst.	⋆ MIST	↑ 1.2	10.6%
		tokenskip	↓ 2.3	13.8%
		gogi_l1	↓ 3.8	7.2%
		perplexity	↓ 14.1	11.1%
		attn_rollout	↓ 2.6	8.0%
		h2o	↓ 6.6	10.0%

Table 1: Performance and compression comparison on GSM8K and MATH-500. Each entry is the mean over $\gamma \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$ relative to the full-chain baseline. ΔAcc. (%): relative accuracy change (↓ drop, ↑ gain); Comp. rate: fraction by which the LoRA-adapted model's inference-time generated tokens are reduced relative to the same full-chain baseline.

where α is a hyperparameter. Tokens ranked by S_i^{MIST} form the compressed chains that supervise the fine-tuning of M at retention budget γ.

4 Experiments

4.1 Experimental Setup

Datasets and Models. We evaluate on four reasoning benchmarks spanning mathematical reasoning (GSM8K (Cobbe et al., 2021); MATH (Hendrycks et al., 2021)) and general-domain reasoning (MMLU-Pro (Wang et al., 2024); BIG-Bench Hard (Suzgun et al., 2023)). Per-dataset split sizes, preprocessing, and self-generation hyperparameters are deferred to Appendix A. We evaluate four models covering three families and three scales: Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct (Team, 2024), Llama-3.1-8B-Instruct (Grattafiori et al., 2024), and Mistral-7B-Instruct-v0.3 (Jiang et al., 2023a).

Baselines. We compare MIST against nine baselines across three categories.

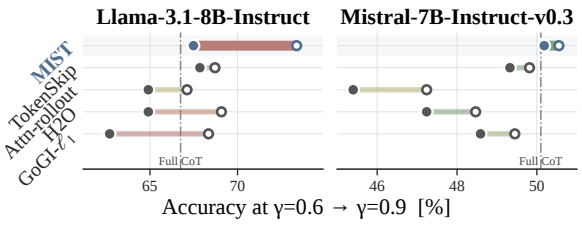

Figure 2: Results on BIG-Bench Hard.

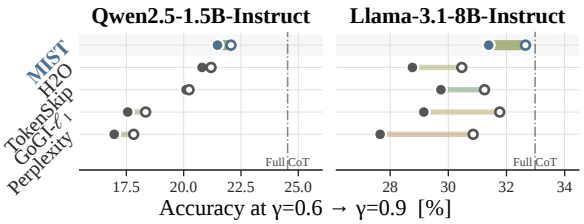

Figure 3: Results on MMLU-Pro.

Training-free baselines: prompt-reduce (Xia et al., 2025) prompts the frozen target model to compress its own chain without finetuning. Naive baselines: full-chain, no-chain, and uniform share the pipeline but use trivial token selection: no compression, chain removal, and uniform retention, respectively. Token scorer baselines: TokenSkip (Xia et al., 2025) scores tokens with the external LLMLingua-2 classifier, GoGI (Zhuang et al., 2025) with the single-layer gradient norm, perplexity with per-token self-information (Jiang et al., 2023b), and attention rollout (Abnar and Zuidema, 2020) and H2O (Zhang et al., 2023) with two attention-based proxies. Per-baseline details are in Appendix B.

Training and evaluation protocol. We adopt the standard procedure from TokenSkip (Xia et al., 2025). Given a target model M and a training set, we (i) sample self-generate reasoning chains from M and keep those that yield the correct answer, (ii) for each retention budget $\gamma \in \{0.5, 0.6, \ldots, 1.0\}$ in a grid, score each chain with MIST and form a compressed chain by retaining the top-$\lceil \gamma T \rceil$ tokens, and (iii) fine-tune M with a single LoRA adapter (rank 8, $\alpha = 16$, learning rate 5×10^{-5}, three epochs, effective batch size 8) on the mixture of compressed chains across all γ values, so that the same adapter can be decoded at any retention budget at inference. It should be noted that since γ controls the supervision tokens retained during SFT rather than imposing a hard length constraint at inference, the adapter is decoded freely and the actual inference-time compression rate (Comp. rate

in Table 1) is typically smaller than $1 - \gamma$. Additional details are provided in Appendix C.

4.2 Main Results

We report main results in Figure 1 and Table 1.

Model-intrinsic saliency outperforms external scorers. TokenSkip is the strongest baseline overall, but it relies on an auxiliary scorer whose importance signal is not derived from the target model's own answer computation. Across the evaluated datasets and models, MIST achieves better performance than TokenSkip. Averaged over γ, MIST incurs only a 1.3-2.4 percentage-point (pp) accuracy drop on GSM8K with Qwen2.5-7B and Mistral-7B, compared with 2.1-4.3 pp for TokenSkip. The gap is even larger on MATH-500: MIST improves average accuracy by 5.3 pp on Qwen2.5-1.5B and 2.3 pp on Llama-3.1-8B, whereas TokenSkip drops by 4.1 pp and 2.3 pp, respectively. This suggests that the target model's own internal computations provide a reliable signal for identifying the reasoning tokens it actually relies on.

A principled internal scorer outperforms internal heuristics. The remaining baselines use target-model signals such as gradients, perplexity, or attention weights, but map these signals to token importance through heuristic scoring rules. For instance, GoGI is the closest internal-gradient baseline, but it treats gradient norm at a preselected layer as a heuristic proxy for token importance, without deriving it from any explicit definition of saliency. MIST, by contrast, operationalizes token importance through necessity and sufficiency interventions and derives its scorer from their first-order effects on the answer likelihood. The gradient appears in MIST only multiplied by the activation and aggregated across all layers; the bare gradient norm used by GoGI can be viewed as a strict simplification of this construction along the necessity axis. The empirical gaps further highlight the advantage of MIST's principled saliency construction. On GSM8K, GoGI incurs 11.9 and 9.2 pp accuracy drops with Qwen2.5-7B and Mistral-7B, respectively, compared with only 2.4 and 1.3 pp for MIST. On MATH-500, GoGI drops by 6.6 pp with Qwen2.5-1.5B and 3.8 pp with Llama-3.1-8B, while MIST improves over full CoT by 5.3 and 1.2 pp. Other internal or model-derived heuristics degrade further, with perplexity, attention rollout, and H2O reaching drops as large as 39.5, 28.8, and 32.6 pp, respectively (Table 1).

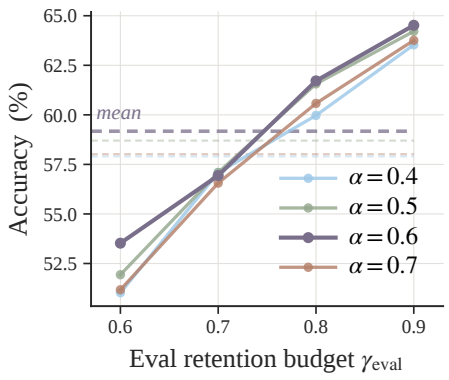

Figure 4: Sensitivity analysis for α on GSM8K with Qwen2.5-1.5B-Instruct.

MIST achieves favorable accuracy-compression trade-off. MIST achieves competitive compression while incurring the smallest reasoning performance loss. On GSM8K with Qwen2.5-7B, MIST achieves a 21.3% reduction in chain length, compared with 20.0% for TokenSkip, while incurring only a 2.4-pp accuracy drop compared with 4.3 pp for TokenSkip. Methods with internal heuristics signals, such as perplexity, attention rollout, and H2O, can achieve similar or higher compression rates but suffer much larger accuracy drops. This trade-off suggests that, under the same retention budget, the tokens selected by MIST carry a greater share of the answer-relevant signal, allowing the adapter to recover chain-level accuracy from a shorter trace.

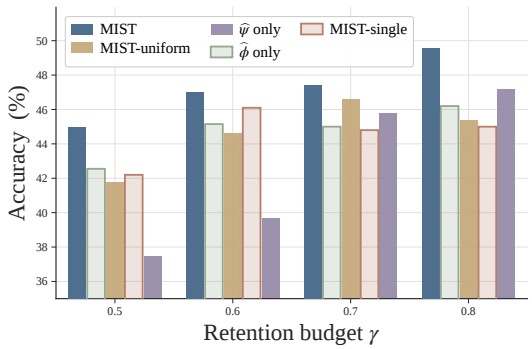

Figure 5: Component ablation of MIST on MATH-500 with Llama-3.1-8B-Instruct.

4.3 Generalization Beyond Mathematical Reasoning

The main results in §4.2 establish MIST's advantage on two mathematical-reasoning benchmarks, GSM8K and MATH-500. We next ask whether this advantage extends beyond mathematical reasoning. To this end, we further evaluate on two general-

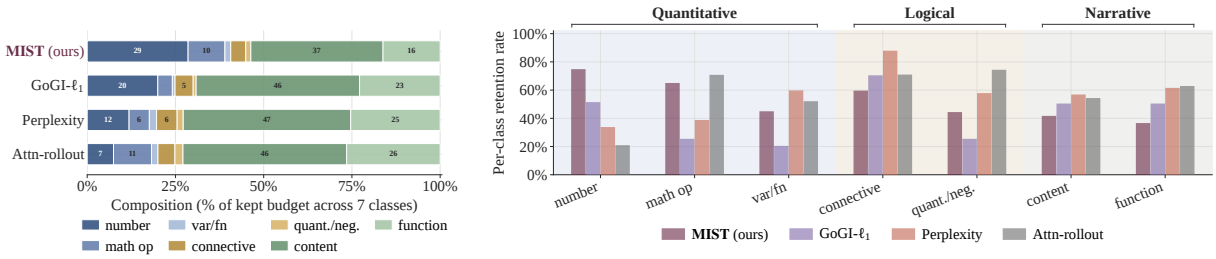

Figure 6: **What each scorer keeps under compression (GSM8K, Qwen2.5-1.5B-Instruct).** Tokens classified using the taxonomy in Figure 7. **Left.** How each method allocates its 50% retention budget across the 7 token types. **Right.** Per-class retention rate: for each class (x-axis), the fraction of class tokens retained by each method.

domain benchmarks: MMLU-Pro and BBH.

As shown in Figures 2 and 3, MIST remains the strongest method on both benchmarks and across the evaluated models. On MMLU-Pro, MIST consistently outperforms external and heuristic baselines, indicating that model-intrinsic saliency remains effective in multi-domain, knowledge-intensive settings. On BBH, MIST also stays closest to, and in some settings surpasses, the full CoT reference, while heuristic baselines fall noticeably behind. These results suggest that MIST is not merely exploiting math-specific token patterns; it effectively captures answer-relevant internal contributions across broader domains and more heterogeneous reasoning formats. Additional results are reported in App. F.

4.4 Ablation Study

Sensitivity analysis. We examine the sensitivity of MIST to the mixing coefficient α in Eq. (8), using Qwen2.5-1.5B-Instruct on GSM8K. Figure 4 compares $\alpha \in \{0.4, 0.5, 0.6, 0.7\}$ across retention budgets. Performance is stable across this range, indicating that the blend is not knife-edge sensitive to α. We fix $\alpha = 0.6$ in all main experiments as it achieves the best overall performance

Each component contributes to the unified MIST quality. We ablate the three design choices that distinguish MIST from one-axis or single-layer simplifications: (i) using both saliency axes, by removing one at a time ($\widehat{\varphi}$-only, $\widehat{\psi}$-only); (ii) the layer weighting in Eq. 7, by replacing $\bar{c}_l$ with a uniform per-layer weight (MIST-uniform); and (iii) the multi-layer aggregation, by computing each axis from only the penultimate layer (MIST-single). Figure 5 reports accuracy on MATH-500 with Llama-3.1-8B-Instruct across $\gamma \in \{0.5, 0.6, 0.7, 0.8\}$. Full MIST achieves the best performance at every retention budget. The single-axis ablations lose the most: $\widehat{\psi}$-only drops by up to 7.5 pp at the most aggressive budget (γ=0.5: 37.5% vs. MIST's 45.0%), and $\widehat{\varphi}$-only loses up to 3.4 pp at γ=0.8 (46.2% vs. 49.6%). MIST-uniform and MIST-single each lose between 0.8 and 4.6 pp depending on γ, confirming that neither the weighting nor the multi-layer aggregation is interchangeable with a uniform-weight or single-layer simplification. The three design choices combine to yield MIST's final quality.

4.5 Token Type Analysis

To understand what each method actually keeps, we partition every token into one of seven role classes grouped into three macro-categories (Figure 7): Quantitative (number, math operator, variable / function), Logical (connective, quantifier / negation), and Narrative (content, function). We then measure each method's budget composition (left) and per-class retention rate (right) on Qwen2.5-1.5B-Instruct + GSM8K at $\gamma = 0.5$ (Figure 6).

Figure 6 reports two complementary views. The left panel shows how each method allocates its retained-token budget across token classes. MIST allocates a larger share of its budget to quantitative tokens than heuristic baselines do: 29% to numbers and 10% to mathematical operators, while still retaining content and function words for local coherence. Within these classes, MIST also retains the highest fraction: 75% of all numbers and 66% of all mathematical operators (Figure 6 right). In contrast, attention rollout and perplexity allocate only 7% and 12% of their budgets to numbers, retaining 22% and 33% of number tokens, respectively.

MIST is therefore the only scorer that both over-allocates budget to Quantitative tokens and achieves the highest within-class retention for them. Attention rollout concentrates on frequent operators and function words that often act as attention hubs, while perplexity tends to conflate surprisal,

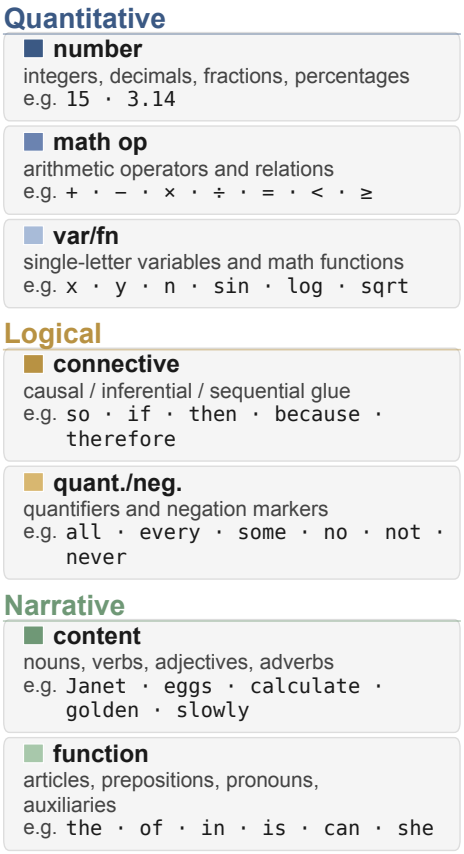

Figure 7: Token taxonomy used in the analysis.

retaining high-surprisal but information-light tokens such as *therefore* and *so*. This pattern is consistent with the diagnosis in §4.2: heuristic scorers concentrate on syntactically prominent but information-light tokens, while MIST's saliency-derived ranking aligns with the tokens that carry the answer-relevant computation.

4.6 Case Study

Figure 8 visualizes MIST's saliency on a three-step GSM8K chain. Necessity $\widehat{\phi}$ assigns high saliency to tokens that the full-chain computation depends on, concentrating on the intermediate numerical facts ($4, 30) and the final answer $120. Sufficiency $\widehat{\psi}$ instead highlights a related but distinct set of tokens, including those associated with buying, spending, currency, and equality, whose residual states carry answer-relevant information when supplied to the no-chain computation. The two views overlap on some answer-bearing tokens but also emphasize different parts of the reasoning trace. Combining these signals, the unified MIST score retains tokens around the arithmetic computation $\$4 \times 30 = \$120$ together with the answer-bearing phrase "spends $120", preserving both quantitative anchors and the local semantic context that binds them, and matching the class-level allocation reported in §4.5.

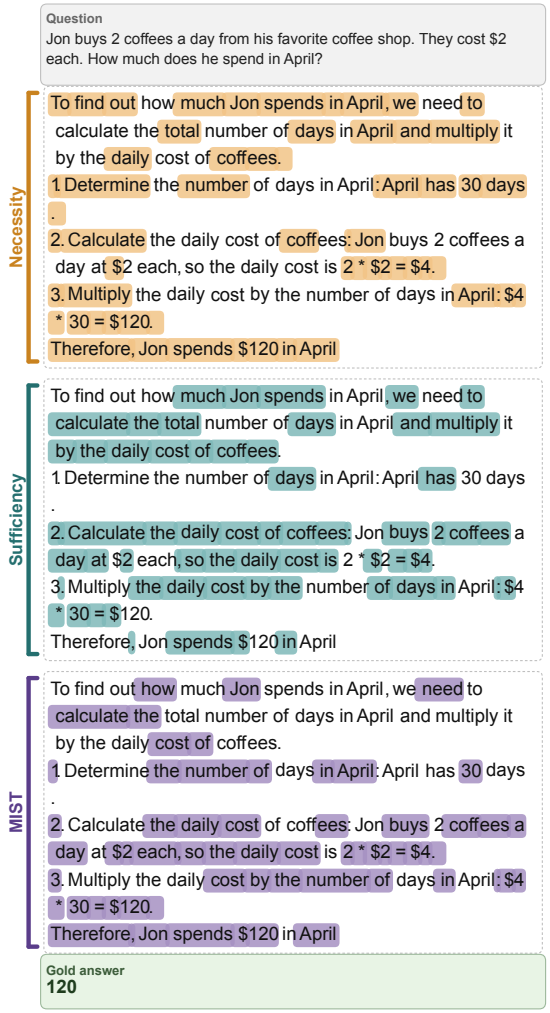

Figure 8: Per-token MIST saliency on a GSM8K chain (Qwen2.5-1.5B-Instruct, $\gamma = 0.5$)

5 Conclusion

We present MIST, a model-intrinsic method for token-level CoT compression. MIST defines token importance operationally through two complementary interventions instantiated on the residual stream, necessity and sufficiency, and converts both into a tractable per-token scorer via first-order Taylor approximations, reducing the cost of token-wise interventions to two backward passes per chain. Extensive empirical evaluations across diverse datasets and models demonstrate the effectiveness of our proposed method.

Limitations

Model scale and gradient access. Our experiments use 1.5B to 8B instruction-tuned models. Whether the saliency signal remains as concentrated at the scale of frontier reasoning models (e.g., 30B to 70B reasoning-tuned LLMs) is not established. Besides, MIST requires access to the target model's internal activations and gradients, so it does not apply to black-box API-only systems.

Evaluation domain. Our four benchmarks span mathematical and general-domain reasoning. Future work can extend the framework to open-ended generation, code generation, or multi-turn dialogue to further validate the effectiveness of the proposed method.

Dependence on correct self-generated chains. Following the TokenSkip pipeline, we score and compress only chains that the target model self-generates correctly. Examples on which the model fails to produce a correct chain are excluded, so our results do not show that MIST repairs incorrect reasoning or improves cases beyond the base model's reach.

Approximate intervention scores. MIST approximates the necessity and sufficiency interventions with their first-order Taylor terms in the residual stream. Because transformer computations are nonlinear, higher-order interactions among tokens, layers, and heads are not captured, so the MIST score is a model-intrinsic saliency estimator rather than an exact causal decomposition.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 4190–4197.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, and 1 others. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12.

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. 2024. Patchscopes: A unifying framework for inspecting hidden representations of language models. In *International Conference on Machine Learning*, pages 15466–15490. PMLR.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2025. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24842–24855.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*.

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

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b. *ArXiv*, abs/2310.06825.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023b. Llmlingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 13358–13376.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Zeju Li, Jianyuan Zhong, Ziyang Zheng, Xiangyu Wen, Zhijian Xu, Yingying Cheng, Fan Zhang, and Qiang Xu. 2026. Making slow thinking faster: Compressing llm chain-of-thought via step entropy. In *The Fourteenth International Conference on Learning Representations*.

Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025. Cot-valve: Length-compressible chain-of-thought tuning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6025–6035.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372.

Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. 2025. Self-training elicits concise reasoning in large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 25127–25152.

nostalgebraist. 2020. interpreting gpt: the logit lens. *LessWrong*.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, and 1 others. 2024. Llmlingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. 2025. Codi: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 677–693.

Janvijay Singh and Dilek Hakkani-Tür. 2026. Do llms encode functional importance of reasoning tokens? *arXiv preprint arXiv:2601.03066*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and 1 others. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051.

Aaquib Syed, Can Rager, and Arthur Conmy. 2024. Attribution patching outperforms automated circuit discovery. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 407–416.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33:12388–12401.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. 2025. Tokenskip: Controllable chain-of-thought compression in llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3351–3363.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*.

Fred Zhang and Neel Nanda. 2024. Towards best practices of activation patching in language models: Metrics and methods. In *The Twelfth International Conference on Learning Representations*.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Re, Clark Barrett, Zhangyang Wang, and Beidi Chen. 2023. H2o: Heavy-hitter oracle for efficient generative inference of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Ren Zhuang, Ben Wang, and Shuifa Sun. 2025. Accelerating chain-of-thought reasoning: When goal-gradient importance meets dynamic skipping. *arXiv preprint arXiv:2505.08392*.

A Dataset Details

Table 2 summarizes the per-dataset split sizes used throughout our evaluation pipeline. Below we describe per-dataset preprocessing and self-generation hyperparameters.

GSM8K. We use the standard `train` and `test` splits from the GSM8K release without further filtering.

Dataset	Train	Test
GSM8K	7,473	1,319
MATH	7,500	500
MMLU-Pro	10,501	1,531
BBH-MC	3,258	816

Table 2: Dataset split sizes used in our evaluation pipeline.

MATH. The training set is the concatenation of the seven Hendrycks (Hendrycks et al., 2021) subjects; the test set is the canonical MATH-500 subset.

MMLU-Pro. The original release does not provide a standard training split, supplying only a 12,032-example test split and a 70-example few-shot validation set. We therefore randomly partition the 12,032 examples into 10,501 training and 1,531 evaluation examples.

BBH-MC. We use the 17 letter-multiple-choice subtasks of BIG-Bench Hard (Suzgun et al., 2023), excluding the 10 binary or free-form subtasks. BBH does not provide a standard training split, supplying only a test set per subtask, so we randomly partition the available examples into 3,258 training and 816 evaluation examples (80/20 split).

Subtask	Letters	N
`date_understanding`	A–F	250
`disambiguation_qa`	A–C	250
`geometric_shapes`	A–K	250
`hyperbaton`	A–B	250
`logical_deduction_three_objects`	A–C	250
`logical_deduction_five_objects`	A–E	250
`logical_deduction_seven_objects`	A–G	250
`movie_recommendation`	A–D	250
`penguins_in_a_table`	A–E	146
`reasoning_about_colored_objects`	A–R	250
`ruin_names`	A–D	250
`salient_translation_error_detection`	A–F	250
`snarks`	A–B	178
`temporal_sequences`	A–D	250
`tracking_shuffled_objects_three_objects`	A–C	250
`tracking_shuffled_objects_five_objects`	A–E	250
`tracking_shuffled_objects_seven_objects`	A–G	250
Total		**4,074**

Table 3: The 17 subtasks of BIG-Bench Hard used in our MIST evaluation. "Letters" shows the answer-letter range; "N" is the original number of examples per subtask before our 80/20 partition.

B Baseline Details

This appendix details the nine baselines compared in Section 4.1.

`Prompt-reduce`. The frozen target model is prompted with the chain-of-thought instruction together with the reduction directive from TokenSkip (Xia et al., 2025, Table 1), which appends `please reduce` $(1-\gamma)\times 100\%$ `of the words in your CoT`. This is a frozen-model, γ-aware baseline that compresses the chain in a single forward pass without any finetuning.

`Full-chain`. The LoRA adapter is finetuned on the unmodified self-generated chains ($\gamma = 1.0$, no token selection).

`No-chain`. The LoRA adapter is finetuned on the self-generated answer span only, with the entire reasoning chain stripped.

`Uniform`. For each chain, a uniformly random subset of $\lceil \gamma T \rceil$ tokens is retained at each retention budget.

`TokenSkip`. We follow Xia et al. (2025) and use a pretrained LLMLingua-2 (Pan et al., 2024) classifier as the external token scorer. The classifier assigns per-token importance values to each self-generated chain, and the top-$\lceil \gamma T \rceil$ tokens are retained at each retention budget.

`GoGI-`ℓ_1. We use the single-layer gradient norm scorer from Adaptive GoGI-Skip (Zhuang et al., 2025). The gradient of $\log p_M(a \mid x, c)$ is taken at the residual state of the last causally connected layer ($l^\star = \max(L-2, 0)$), and the per-token score is $\|\nabla_{h_i^{(l^\star)}} \log p_M\|_1$. We adopt the single-layer formulation as the closest gradient-norm-only counterpart to MIST.

`Perplexity`. We use per-token self-information $s_i = -\log p_M(t_i \mid x, t_{<i})$ following the LLMLingua-1 prompt compression scorer (Jiang et al., 2023b); higher-self-information tokens are retained, on the grounds that they carry more conditional information.

`Attention rollout`. Following Abnar and Zuidema (2020), we aggregate per-head attention weights into a cross-layer importance score. Starting from the identity, the cumulative attention matrix at layer l is

$$A^{(l)} = (\widetilde{A}^{(l)} + I)\, A^{(l-1)}, \quad A^{(0)} = I, \qquad (9)$$

where $\widetilde{A}^{(l)}$ is the head-averaged attention matrix at layer l. The per-token score is the column sum of $A^{(L)}$ at the answer position(s).

H2O. We use the Heavy-Hitter Oracle score (Zhang et al., 2023), defined as the cumulative attention mass received by each token across all layers and heads from the answer position(s):

$$\text{score}(i) = \sum_{l,h} \sum_{j \in \text{answer}} A_{j,i}^{(l,h)}. \tag{10}$$

C Training, Evaluation, and Implementation Details

This appendix details the pipeline, hyperparameters, evaluation protocol, and hardware configuration used in all experiments.

Pipeline overview. For each (model, dataset) cell we follow a three-stage protocol: (i) self-generate CoT chains from the target model on the training set and retain those whose extracted answer matches the gold label; (ii) score each retained chain with MIST or the baseline scorer, producing per-token importance values; and (iii) for each retention budget $\gamma \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$, retain the top-$\lceil \gamma T \rceil$ tokens of every chain, fine-tune the target model with a LoRA adapter on the resulting compressed chains, and evaluate by greedy decoding on the test set.

We distinguish two ratios that reviewers may otherwise conflate. The training retention budget γ is the per-chain fraction of CoT tokens kept when constructing the SFT supervision (so $\gamma = 0.7$ retains 70% of the original chain's tokens). The reported inference compression rate is the relative reduction in total tokens generated by the fine-tuned adapter, measured against the full-chain LoRA baseline. The two need not coincide: the adapter learns its own length policy from the γ-mixture and, especially on smaller models, often produces chains shorter than the average retention floor would suggest.

Self-generation (Phase 0). We greedy-decode the target model on training-set queries using temperature 0 and top-p 1.0. The maximum number of new tokens per chain is set per dataset to match each task's typical chain length: 512 for GSM8K, 768 for MMLU-Pro and BBH-MC, and 1,024 for MATH. Generations are filtered to keep only chains whose extracted answer matches the gold label. Self-generation uses the SDPA attention backend and `bfloat16` precision for numerical stability with batched generation (batch size 32).

Reproducibility and fairness. The same set of self-generated chains, sampled once per (model, dataset) cell, is reused across all scorers (MIST and all baselines), so any performance difference is attributable to the scoring stage rather than to differences in supervision data.

Token scoring (Phase 1). For each retained chain, computing MIST requires two backward passes through the target model: a source backward through the answer log-likelihood $\log p_M(a \mid x, c)$ on the full prompt-and-chain input, and a target backward through $\log p_M(a \mid x, \emptyset)$ on the prompt-only no-chain input. We register retain-grad hooks at the output of every transformer block, so a single source backward simultaneously exposes $\nabla_{h_i^{(l)}} \log p_M^{\text{src}}$ for every chain position i and every layer l (Appendix D.4); the target backward analogously exposes the per-layer gradient at the target's final position. The per-layer activations $h_i^{(l),\text{src}}$ and the logit-lens layer weight $\bar{c}_l$ are read off the same source forward pass. We use the eager attention backend and `float32` precision for this stage to preserve gradient numerical stability.

LoRA fine-tuning (Phase 2). We follow the TokenSkip (Xia et al., 2025) fine-tuning recipe so that any performance gap between MIST and a baseline scorer can be attributed to the scoring stage alone. LoRA adapters use rank $r = 8$, scaling $\alpha = 16$, dropout 0.0, and target all linear projections (`q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj`). Training uses learning rate 5×10^{-5} with linear warmup over 10% of total steps, 3 epochs, per-device batch size 1, gradient accumulation 8 (effective batch 8), and context length 2,048. For each (model, dataset, scorer) triple we train a single LoRA adapter on a uniform mixture of compressed chains at $\gamma \in 0.5, 0.6, 0.7, 0.8, 0.9, 1.0$, following TokenSkip's γ-mixing recipe (Xia et al., 2025). At inference, the same adapter is decoded under each evaluation γ by prepending the standard `Please reduce` $(1-\gamma) \times 100\%$ `of the words in your CoT` directive (TokenSkip Table 1); no separate per-γ adapter is trained. The adapter is saved once per triple to `adapters/<run_key>/` for evaluation reuse.

Evaluation (Phase 3). At evaluation, each adapter is loaded into the target model and generation is performed by greedy decoding (temperature 0, top-p 1.0) on the held-out test set with evaluation batch size 16. The maximum number of new tokens per generation matches the Phase

0 budget for each dataset (512 for GSM8K, 768 for MMLU-Pro and BBH-MC, 1,024 for MATH). We use the SDPA attention backend and `bfloat16` precision, matching Phase 0 for consistency. Per-batch wall-clock decoding time is recorded for the inference-cost analysis reported in §4.2.

Hardware and software. All experiments run on 2× NVIDIA A100 (80 GB) GPUs. The implementation uses PyTorch 2.9, Hugging Face Transformers, and the PEFT library for LoRA fine-tuning. The pipeline incurs the following one-time cost: self-generation of reasoning chains takes 0.5–2 GPU-hours; scoring chains with MIST requires two backward passes per chain and takes 0.3–1.0 GPU-hours; LoRA adapter training takes 1–3 GPU-hours; and evaluation at five retention budgets takes 0.5–1.5 GPU-hours. Aggregated across all four models, four datasets, and ten methods (six scorers plus four trivial baselines), the total compute reported in this paper is on the order of 1,500 GPU-hours.

D Taylor Expansions and Remainder Bounds

This appendix formalizes the first-order Taylor expansions of the per-layer necessity and sufficiency terms $\widehat{\phi}_i^{(l)}$ and $\widehat{\psi}_i^{(l)}$ and gives explicit per-layer remainder bounds, completing the technical content deferred from Section 3.2

D.1 Setup and notation

Local per-layer functions. Let M be a transformer language model with L residual-stream layers and hidden dimension d. For each chain position $i \in [T]$ and layer $l \in [L]$, define the source-side local function

$$f_{i,l}^{\mathrm{src}}(\mathbf{u}) := \log p_M\big(a \mid x, c;\ h_i^{(l)} \leftarrow \mathbf{u}\big), \quad \mathbf{u} \in \mathbb{R}^d \tag{11}$$

where the semicolon notation indicates that the residual-stream state $h_i^{(l)}$ is replaced by $\mathbf{u}$, after which the remaining computation is evaluated normally. At the natural value the intervened computation coincides with the standard forward pass, and its derivative with respect to $\mathbf{u}$ coincides with the reverse-mode gradient of $\log p_M^{\mathrm{src}}$ with respect to $h_i^{(l)}$:

$$\nabla f_{i,l}^{\mathrm{src}}\big(h_i^{(l),\mathrm{src}}\big) \;=\; \nabla_{h_i^{(l)}} \log p_M^{\mathrm{src}}. \tag{12}$$

The target-side local function is defined analogously:

$$f_l^{\mathrm{tgt}}(\mathbf{u}) \;:=\; \log p_M\big(a \mid x, \emptyset;\ h_{\mathrm{final}}^{(l)} \leftarrow \mathbf{u}\big), \quad \mathbf{u} \in \mathbb{R}^d, \tag{13}$$

with the analogous gradient identity

$$\nabla f_l^{\mathrm{tgt}}\big(h_{\mathrm{final}}^{(l),\mathrm{tgt}}\big) \;=\; \nabla_{h_{\mathrm{final}}^{(l)}} \log p_M^{\mathrm{tgt}}. \tag{14}$$

For brevity, write $\log p_M^{\mathrm{src}} := \log p_M(a \mid x, c)$ and $\log p_M^{\mathrm{tgt}} := \log p_M(a \mid x, \emptyset)$.

Regularity assumption. We assume each $f_{i,l}^{\mathrm{src}}$ and f_l^{tgt} is locally C^2 on a neighborhood containing the unperturbed point and the corresponding perturbed point. This is a working assumption invoked only to state Lagrange-form Taylor remainders; the empirical method requires only first-order gradients.

D.2 Necessity: Taylor expansion of $\phi_i^{(l)}$

The per-layer necessity intervention at layer l replaces $h_i^{(l)}$ with $\mathbf{0}$ and lets subsequent layers evaluate normally from this perturbation. The corresponding per-layer saliency and displacement are

$$\begin{aligned} \phi_i^{(l)} &:= f_{i,l}^{\mathrm{src}}\big(h_i^{(l),\mathrm{src}}\big) - f_{i,l}^{\mathrm{src}}\big(\mathbf{0}\big), \\ \delta_{i,l}^{\phi} &:= -\,h_i^{(l),\mathrm{src}} \in \mathbb{R}^d. \end{aligned} \tag{15}$$

Expansion. By Taylor's theorem with Lagrange remainder applied to $f_{i,l}^{\mathrm{src}}$ along the segment from $h_i^{(l),\mathrm{src}}$ to $\mathbf{0}$,

$$\begin{aligned} f_{i,l}^{\mathrm{src}}(\mathbf{0}) = f_{i,l}^{\mathrm{src}}\big(h_i^{(l),\mathrm{src}}\big) &+ \big\langle \nabla f_{i,l}^{\mathrm{src}}\big(h_i^{(l),\mathrm{src}}\big),\ \delta_{i,l}^{\phi} \big\rangle \\ &+ \tfrac{1}{2}\big(\delta_{i,l}^{\phi}\big)^{\top} \nabla^2 f_{i,l}^{\mathrm{src}}\big(\xi_{i,l}^{\phi}\big)\delta_{i,l}^{\phi}. \end{aligned} \tag{16}$$

for some $\xi_{i,l}^{\phi}$ on the segment. Substituting $\delta_{i,l}^{\phi} = -h_i^{(l),\mathrm{src}}$, using Eq. (12), and rearranging yields

$$\phi_i^{(l)} \;=\; \widehat{\phi}_i^{(l)} \;-\; R_{i,l}^{\phi} \tag{17}$$

where

$$\widehat{\phi}_i^{(l)} \;=\; \big\langle \nabla_{h_i^{(l)}} \log p_M^{\mathrm{src}},\ h_i^{(l),\mathrm{src}} \big\rangle, \tag{18}$$

$$R_{i,l}^{\phi} \;=\; \tfrac{1}{2}\,\big(\delta_{i,l}^{\phi}\big)^{\top} \nabla^2 f_{i,l}^{\mathrm{src}}\big(\xi_{i,l}^{\phi}\big)\delta_{i,l}^{\phi}. \tag{19}$$

Eq. (17) recovers the per-layer first-order term defined in Eq. (4) of the main text.

Remainder bound. Let

$$\beta^{\phi}_{i,l} := \sup_{\xi \in \mathcal{S}^{\phi}_{i,l}} \|\nabla^2 f^{\mathrm{src}}_{i,l}(\xi)\|_{\mathrm{op}} \tag{20}$$

denote the Hessian operator-norm bound on the segment $\mathcal{S}^{\phi}_{i,l}$ from $h^{(l),\mathrm{src}}_i$ to $\mathbf{0}$. Then

$$\left|R^{\phi}_{i,l}\right| \;\le\; \tfrac{1}{2}\,\beta^{\phi}_{i,l}\,\left\|h^{(l),\mathrm{src}}_i\right\|_2^2. \tag{21}$$

D.3 Sufficiency: Taylor expansion of $\psi^{(l)}_i$

The per-layer sufficiency intervention at layer l replaces the target's final-position residual at layer l with the source's position-i residual at the same layer, and lets subsequent layers evaluate normally from this perturbation. The corresponding per-layer saliency and displacement are

$$\begin{aligned} \psi^{(l)}_i &:= f^{\mathrm{tgt}}_l\big(h^{(l),\mathrm{src}}_i\big) - f^{\mathrm{tgt}}_l\big(h^{(l),\mathrm{tgt}}_{\mathrm{final}}\big), \\ \delta^{\psi}_{i,l} &:= h^{(l),\mathrm{src}}_i - h^{(l),\mathrm{tgt}}_{\mathrm{final}} \in \mathbb{R}^d. \end{aligned} \tag{22}$$

Expansion. By Taylor's theorem with Lagrange remainder applied to f^{tgt}_l along the segment from $h^{(l),\mathrm{tgt}}_{\mathrm{final}}$ to $h^{(l),\mathrm{src}}_i$,

$$\begin{aligned} f^{\mathrm{tgt}}_l\big(h^{(l),\mathrm{src}}_i\big) = f^{\mathrm{tgt}}_l\big(h^{(l),\mathrm{tgt}}_{\mathrm{final}}\big) &+ \big\langle \nabla f^{\mathrm{tgt}}_l\big(h^{(l),\mathrm{tgt}}_{\mathrm{final}}\big),\, \delta^{\psi}_{i,l}\big\rangle \\ &+ \tfrac{1}{2}\big(\delta^{\psi}_{i,l}\big)^{\top}\nabla^2 f^{\mathrm{tgt}}_l\big(\xi^{\psi}_{i,l}\big)\delta^{\psi}_{i,l}. \end{aligned} \tag{23}$$

for some $\xi^{\psi}_{i,l}$ on the segment. Using Eq. (14) and rearranging yields

$$\psi^{(l)}_i \;=\; \widehat{\psi}^{(l)}_i \;+\; R^{\psi}_{i,l} \tag{24}$$

where

$$\widehat{\psi}^{(l)}_i \;=\; \big\langle \nabla_{h^{(l)}_{\mathrm{final}}} \log p^{\mathrm{tgt}}_M,\; h^{(l),\mathrm{src}}_i - h^{(l),\mathrm{tgt}}_{\mathrm{final}}\big\rangle, \tag{25}$$

$$R^{\psi}_{i,l} \;=\; \tfrac{1}{2}\,\big(\delta^{\psi}_{i,l}\big)^{\top}\nabla^2 f^{\mathrm{tgt}}_l\big(\xi^{\psi}_{i,l}\big)\delta^{\psi}_{i,l}. \tag{26}$$

Eq. (24) recovers the per-layer first-order term defined in Eq. (5) of the main text. The plus sign on $R^{\psi}_{i,l}$ (as opposed to the minus sign on $R^{\phi}_{i,l}$) appears because sufficiency is defined as a likelihood gain whereas necessity is defined as a likelihood drop.

Remainder bound. Let

$$\beta^{\psi}_{i,l} := \sup_{\xi \in \mathcal{S}^{\psi}_{i,l}} \|\nabla^2 f^{\mathrm{tgt}}_l(\xi)\|_{\mathrm{op}} \tag{27}$$

denote the Hessian operator-norm bound on the segment $\mathcal{S}^{\psi}_{i,l}$. Then

$$\left|R^{\psi}_{i,l}\right| \;\le\; \tfrac{1}{2}\,\beta^{\psi}_{i,l}\,\left\|h^{(l),\mathrm{src}}_i - h^{(l),\mathrm{tgt}}_{\mathrm{final}}\right\|_2^2. \tag{28}$$

D.4 Recovering all per-layer gradients from a single backward pass

Computing $\widehat{\phi}^{(l)}_i$ for every (i,l) pair naively suggests TL separate backward passes. In practice, a single backward through $\log p^{\mathrm{src}}_M$ simultaneously recovers all gradients $\nabla_{h^{(l)}_i} \log p^{\mathrm{src}}_M$ by registering retain-grad hooks on each transformer block's output residual; reverse-mode autograd propagates gradients to all intermediate activations through the standard chain rule. The hooks expose these gradients without modifying the computation graph. Thus all necessity terms require one source forward and one backward pass through M, and all sufficiency terms require one target forward and one backward pass, each up to activation-caching overhead.

D.5 Empirical Tightness of the First-Order Term

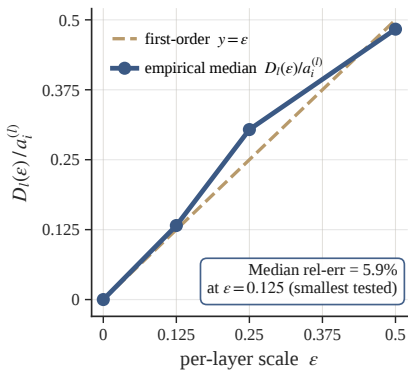

Figure 9: Empirical tightness of the per-layer first-order Taylor term.

The analytic bounds in §D.2–D.3 are worst-case statements parametrised by an unspecified Hessian constant; we close that gap by measuring directly how tightly the first-order term predicts the true ablation drop on real chains. For one chain token i at one residual block l, we scale the residual stream $h^{(l)}_i \to (1-\varepsilon)\,h^{(l)}_i$ at that block only (others unperturbed) and measure the answer-log-likelihood drop

$$D_l(\varepsilon) := \log p_M(a\,|\,x,c) - \log p_M\big(a\,|\,x,c;\; h^{(l)}_i \leftarrow (1-\varepsilon)\,h^{(l)}_i\big). \tag{29}$$

The per-layer Taylor main term (Eq. 4), $a^{(l)}_i := \langle \nabla_{h^{(l)}_i} \log p^{\mathrm{src}}_M,\, h^{(l)}_i\rangle = \widehat{\phi}^{(l)}_i$, yields the first-order prediction $D_l(\varepsilon) \approx \varepsilon a^{(l)}_i$, so the normalized drop $D_l(\varepsilon)/a^{(l)}_i$ equals ε in the first-order-tight limit. Measured on 8 GSM8K chains with Qwen2.5-1.5B-Instruct in fp32 over functionally active (i,l) cells

(top quartile of $|D_l(\varepsilon = 1)|$ per chain, floored at 10^{-5} nats), the empirical median tracks $y = \varepsilon$ within $\mathbf{5.9}\%$ at $\varepsilon = 0.125$ and $\lesssim 10\%$ overall(Figure 9).

E Licenses and Terms of Use

This section discusses the licenses and terms of use for all external artifacts used in our experiments and for the artifacts we release. All datasets and models we use are publicly available under permissive research-friendly licenses, and our use is consistent with their intended research purposes.

Datasets. We use four publicly released reasoning benchmarks. GSM8K (Cobbe et al., 2021) is released under the MIT License. The MATH dataset (Hendrycks et al., 2021) is released under the MIT License. MMLU-Pro (Wang et al., 2024) is released under the MIT License via the TIGER-Lab repository on Hugging Face. BIG-Bench Hard (Suzgun et al., 2023) is released under the Apache License 2.0 as part of the BIG-Bench suite. All four datasets are distributed for research use and our experimental use is consistent with their intended scope. We do not redistribute the underlying data; experiments load the datasets from their original sources.

Models. We use four publicly released instruction-tuned models. Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct (Team, 2024) are released under the Qwen License (Apache 2.0–based). Llama-3.1-8B-Instruct (Grattafiori et al., 2024) is released under the Llama 3.1 Community License Agreement. Mistral-7B- Instruct-v0.3 (Jiang et al., 2023a) is released under the Apache License 2.0. All four models permit research use, fine-tuning, and the release of derived artifacts (such as LoRA adapters) under their respective terms.

Consistency with intended use. Our use of all external artifacts is consistent with their intended research purposes. The four reasoning datasets (GSM8K, MATH, MMLU-Pro, BIG-Bench Hard) are released as research benchmarks for evaluating multi-step reasoning in language models, and our use is restricted to benchmarking and training LoRA adapters on the released training splits. The four instruction-tuned models (Qwen2.5-1.5B/7B, Llama-3.1-8B, Mistral-7B-Instruct-v0.3) are released to support research, fine-tuning, and the development of derived artifacts under their respective terms. Adapters derived from each base model inherit the original access conditions of that base model; in particular, adapters derived from Llama-3.1-8B-Instruct are released only for use with the Llama-3.1 base model, and are not used to train, distill into, or otherwise improve language models outside the Llama family, consistent with the Llama 3.1 Community License. All other adapters (built on Qwen2.5 and Mistral, both released under Apache 2.0) carry no such derivative restrictions. All experimental use is for research purposes only; we do not deploy the trained adapters in commercial products.

Dataset	Model	Method	Δ Acc. (%)	Comp. rate
BBH	Llama-3.1-8B-Inst.	⋆ MIST	↑ 3.7	11.6%
		tokenskip	↑ 1.5	4.6%
		gogi_l1	↓ 1.2	6.8%
		attn_rollout	↓ 0.7	6.1%
		h2o	↑ 0.2	7.8%
MMLU-Pro	Qwen2.5-1.5B-Inst.	⋆ MIST	↓ 2.8	20.8%
		tokenskip	↓ 4.3	−0.5%
		gogi_l1	↓ 6.6	3.4%
		perplexity	↓ 7.1	7.8%
		h2o	↓ 3.5	12.7%
	Llama-3.1-8B-Inst.	⋆ MIST	↓ 1.0	3.4%
		tokenskip	↓ 2.5	1.2%
		gogi_l1	↓ 2.6	2.1%
		perplexity	↓ 3.7	6.9%
		h2o	↓ 3.4	2.5%

Table 4: Additional main results on BBH and MMLU-Pro, in the same format as Table 1. Each entry is the mean over $\gamma \in \{0.6, 0.7, 0.8, 0.9\}$ relative to the Full CoT baseline. ΔAcc. (%): relative accuracy change (↓ drop, ↑ gain); Comp. rate: fraction by which the LoRA-adapted model's inference-time generated tokens are reduced relative to the same Full CoT baseline. A negative compression rate indicates that the compressed-supervision adapter generated *more* tokens than the Full CoT baseline at evaluation.

F Additional Main Results on BBH and MMLU-Pro

Table 4 reports the per-(model, scorer) compression numbers underlying Figs. 2 and 3, in the same ΔAcc / compression-rate format as Table 1.

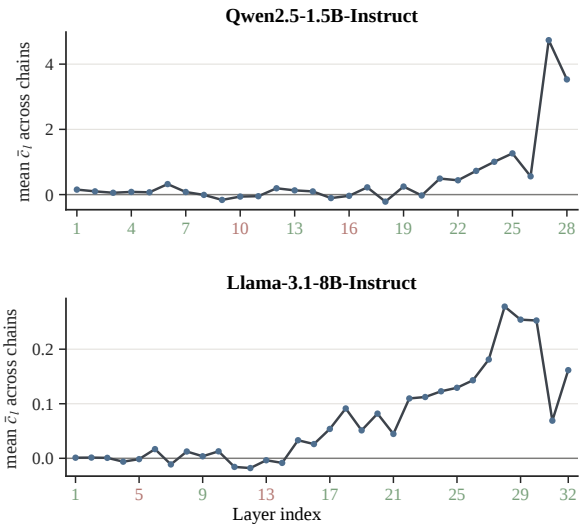

Figure 10: Empirical mean of the logit-lens layer weight $\bar{c}_l$ (Eq. 6) across 30 BBH chains for two of the paper-headline target models. Weight magnitude is overwhelmingly concentrated in the late layers, while many middle layers carry near-zero weight; both observations are direct empirical support for the layer-weighted aggregation in Eq. (7) over a uniform per-layer weight.

G Empirical Per-Layer Logit-Lens Weight

The layer weight $\bar{c}_l$ in Eq. (6) is the inner product of layer l's residual-stream update with the answer unembedding direction, chain-averaged over the chain tokens. Figure 10 plots its mean across 30 BBH chains layer by layer for Qwen2.5-1.5B-Instruct and Llama-3.1-8B-Instruct, two of the four paper-headline target models.

The figure makes the following empirical points concrete. First, $\bar{c}_l$ is far from uniform across depth. This explains why the MIST-uniform ablation in §4.2 (Fig. 5) loses up to 4.6 pp — uniform weighting wastes selection budget on layer-token cells whose updates are essentially answer-neutral. Besides, the late-layer dominance is consistent across the two model families, so the aggregation in Eq. (7) is not specific to a single architecture.

H Per-Axis Normalization Before Combining

The necessity and sufficiency scores $\widehat{\phi}_i$ and $\widehat{\psi}_i$ (Eq. 7) are inner products of log-likelihood gradients with different reference activations and therefore live on different scales across chains. To make the mixing coefficient α in Eq. (8) comparable across chains and benchmarks, we apply a per-chain standardization to each axis before blending.

The necessity axis is standardized in log space:

$$\widehat{\phi}_i^{\text{norm}} = \frac{\log(|\widehat{\phi}_i| + \varepsilon) - \text{mean}_j \log(|\widehat{\phi}_j| + \varepsilon)}{\text{std}_j \log(|\widehat{\phi}_j| + \varepsilon)}, \tag{30}$$

and the sufficiency axis with a standard z-score:

$$\widehat{\psi}_i^{\text{norm}} = \frac{\widehat{\psi}_i - \text{mean}_j \widehat{\psi}_j}{\text{std}_j \widehat{\psi}_j}, \tag{31}$$

with $\varepsilon = 10^{-12}$ for numerical stability and statistics taken over the chain's token indices j. The unified score in Eq. (8) is computed on $\widehat{\phi}_i^{\text{norm}}$ and $\widehat{\psi}_i^{\text{norm}}$. The log transformation on the necessity axis is a standard choice for quantities whose per-chain magnitudes span several orders of magnitude, and preserves the relative ordering of tokens within a chain.