Can Aha Moments be Fake? Towards Quantifying Decorative and True Thinking in Chain-of-Thought

Anonymous ACL submission

Abstract

Large language models can generate long chain-of-thought (CoT) reasoning, yet prior work suggests that CoT can be post-hoc rationalization rather than a faithful reflection of the computation through explicitly designed settings. In this work, we go further and propose a True Thinking Score (TTS) to quantify the causal contribution of each step in CoT to the model's final prediction in realistic reasoning problems. Across eleven models ranging from 1.5B to 1.1T parameters on common reasoning benchmarks, we find that CoTs often interleave *true-thinking* steps, which causally affect the final answer, with *decorative-thinking* steps, which appear useful but have little causal influence; Such decorative steps remain prevalent even for frontier models: Over 30% of steps in Kimi-K2.6 are decorative on MATH with TTS ≤ 0.005. Furthermore, TTS enables effective CoT pruning: removing 50% of CoT steps with the lowest TTS can largely maintain the performance. Self-training on these pruned CoTs reduces reasoning length by 66% while preserving performance on Nemotron3-Nano-30B. Finally, we provide a mechanistic analysis showing that LLMs can be steered in the latent space to engage or disengage with reasoning steps. Overall, our results reveal that frontier LLMs often verbalize reasoning steps that are not causally used, challenging both the efficiency and the trustworthiness of CoT.

1 Introduction

Recent frontier LLMs can increasingly solve complex reasoning problems through test-time scaling, often by generating a very long chain of thought (CoT) (Guo et al., 2025; Muennighoff et al., 2025; Snell et al., 2024; Jaech et al., 2024). In their long CoT, these models frequently exhibit an "aha moment", where the model begins to *self-verify* its solution (Guo et al., 2025). LLMs' generated CoT is commonly assumed as a scratch pad where the model thinks out loud (Korbak et al., 2025). This has also motivated using CoT as a means to monitor LLMs and detect unsafe behaviors revealed in their CoT (Baker et al., 2025).

A central assumption behind CoT is faithfulness: the verbalized reasoning should reflect the computation that produces the model's final answer. Prior work shows that this assumption can fail. Models may rely on inserted hints (Chen et al., 2025b; Chua and Evans, 2025; Turpin et al., 2023) or spurious biases (Arcuschin et al., 2025) without acknowledging them in their CoT, and may commit to an answer before completing the reasoning trace (Ma et al., 2025; Wang et al., 2025; Yang et al., 2025b). These findings suggest that CoT steps can be post-hoc rationalizations rather than causally used parts of the prediction process.

However, prior work largely relies on controlled settings and remains suggestive, typically showing CoT can be unfaithful at the level of the entire reasoning trace. It still remains unclear how each individual CoT step truly contributes to the model's final prediction in common reasoning tasks. This gap limits the use of CoT as a monitoring signal. If a reasoning step does not causally affect the prediction, inspecting it may give a misleading view of the model's computation, especially in high-stakes settings (Zhong et al., 2025; Baker et al., 2025).

To close this gap, we propose the True-Thinking Score (TTS), a step-wise causal analysis of CoT that quantifies how much each verbalized step affects the model's prediction. We define **true-thinking** steps as those that causally affect the prediction, and **decorative-thinking** steps as those that make minimal causal contribution to the prediction, as illustrated in Figure 1. TTS is adapted from Average Treatment Effect (ATE) (Rubin, 1974; Pearl, 2009) with two complementary interventions by conditioning on its context steps to evaluate its necessity and sufficiency to the model's prediction (Section 2.1). On common reasoning benchmarks, we evaluate eight frontier models

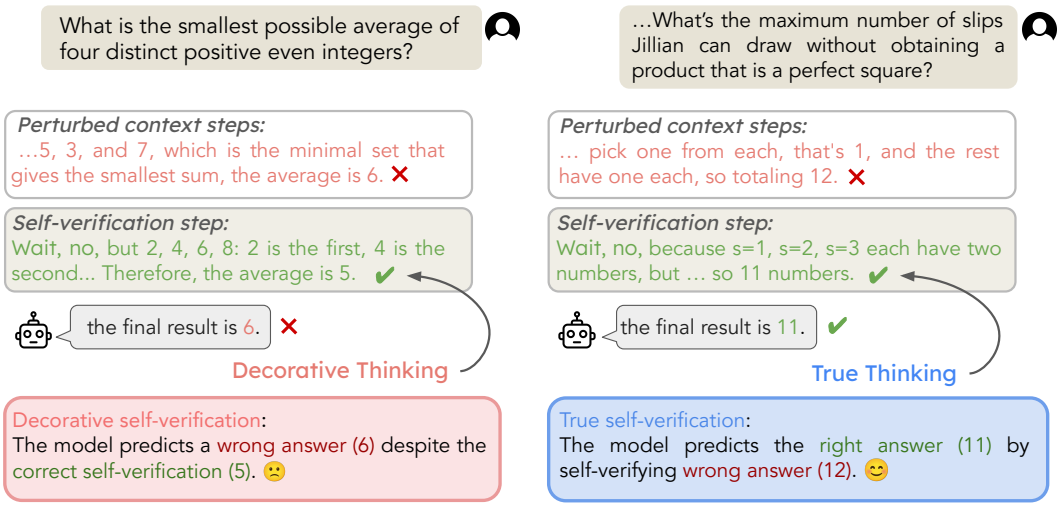


Figure 1: We find that many CoT steps are not *true thinking*, but act as *decorative thinking*: they contribute little causally to the model's answer. Consider self-verification steps. We randomly perturb numerical values in the preceding context and re-prompt the model to answer using the modified CoT. In the left example, although the self-verification step is correct, the model ignores it and outputs the wrong answer after perturbation. In the right example, the model follows the self-verification step and recovers the correct answer. To quantify this distinction, we propose the True-Thinking Score, which measures the causal contribution of each reasoning step to the model's final prediction.

across different scales and find that decorative steps remain widespread (Section 4). On MATH benchmark, nearly 60% of Qwen3.6-35B's CoT steps have TTS $\leq$ 0.005, indicating minimal causal impact on the final answer. Larger models tend to achieve higher TTS, but still contain many decorative steps: for Nemotron3-Super-120B, nearly 30% of steps are decorative. Moreover, true-thinking and decorative-thinking steps are often interleaved within the same CoT, raising concerns about using CoT as a monitoring signal for model misbehavior (Baker et al., 2025).

We next show that TTS enables effective CoT pruning (Section 4.1). Removing the 50% lowest-TTS steps largely preserves performance, while removing high-TTS steps sharply degrades reasoning accuracy. TTS-based pruning also outperforms an expert prompt-compression LM and direct prompting for concise reasoning. Further self-training on pruned CoTs reduces reasoning length by 66% while preserving performance, and generalizes to held-out dataset.

Finally, we provide a mechanistic study of step-level causal use (Section 5). We demonstrate that steering the hidden states of a reasoning step can mediate whether the model uses that step to compute the answer. The steering direction is extracted from the contrast between low- and high-TTS steps on one dataset, yet generalizes to other datasets and appears to modulate the model's attention to the targeted step. In particular, steering along the TrueThinking direction can induce the model to reason over otherwise decorative self-verification steps and correct previous errors.

Overall, our core contributions are threefold.

- We propose TTS to quantify the causal contribution of each step in a CoT. We reveal that decorative thinking steps—steps with minimal causal contribution—are prevalent in frontier models on common reasoning benchmarks.
- We show that TTS can be used to prune CoTs without degrading performance, and that further self-training on pruned CoTs improves reasoning efficiency.
- We provide a mechanistic study, revealing for the first time that whether the model uses a reasoning step to compute the answer can be mediated through steering in the latent space.

Together, these findings call for more future work on improving the causality of LLM reasoning, making CoTs more trustworthy and efficient for real-world deployment.

2 Measuring Step-wise Causality in Reasoning

In this section, we adapt the Average Treatment Effect (ATE) (Rubin, 1974; Pearl, 2009) to

measure the causal impact of individual steps s on LLMs' prediction result y^*. We define that a step with genuine causal impact is a **true-thinking step**, where the model follows s in order to produce y^*. By contrast, a step with minimal causal impact on y^* is a **decorative-thinking step**. We first detail the notations and explain the True-Thinking Score in Section 2.1.

Notation and Setup. We adopt notation following Rubin (1974); Pearl (2009). Let the input question be q, and let the model's full chain of thought (CoT) for q be $\mathcal{C}^\star = (s_1, s_2, \ldots, s_n)$, where each s_i denotes a reasoning step. At the current step s under evaluation (we omit the index i and directly use s for simplicity), we define the **context** as its preceding steps, i.e., $\mathcal{C} = (s_1, s_2, \ldots, s_{i-1})$. To probe the model's current prediction after any partial reasoning trace, we use early-exit answering by appending a standardized cue: The final result is. This approach, following Lanham et al. (2023); Fu et al. (2025); Yang et al. (2025b); Tanneru et al. (2024); Bogdan et al. (2025), reliably elicits the model's intermediate answer given the question q and reasoning prefix $(\mathcal{C}, s)$. Let $f(q, \mathcal{C}, s)$ denote the model's early-exit prediction after processing q with context $\mathcal{C}$ and step s. The *reference prediction* under the full reasoning trace is then defined as $y^* := f(q, \mathcal{C}^\star)$, representing the model's final answer when all steps in the full CoT are intact.

Indicator Variables. We introduce the following binary random variables to formalize interventions on reasoning steps: (1) **Context indicator** $\mathbf{C} \in \{0, 1\}$: $\mathbf{C}{=}1$ indicates an intact context (the original prefix $\mathcal{C}$); $\mathbf{C}{=}0$ indicates a perturbed context in which all preceding steps are replaced by perturbed versions. We write $c \in \{0, 1\}$ for a specific realization of $\mathbf{C}$. (2) **Step toggle** $\mathbf{X} \in \{0, 1\}$: $\mathbf{X}{=}1$ is the original step s; $\mathbf{X}{=}0$ replaces it with a perturbed version s'. (3) **Outcome indicator** $\mathbf{Y} \in \{0, 1\}$: Given $f(q, \mathcal{C}, s)$, we define $\mathbf{Y} := \mathbf{1}\{f(q, \mathcal{C}, s) = y^*\}$, which measures whether the model's early-exit prediction under the given intervention matches the full-CoT reference outcome.

Perturbation Procedure. To isolate the causal effect of each reasoning step, we create perturbed versions of steps and contexts by introducing *small random numerical offsets* to quantities appearing in the reasoning text (Lanham et al., 2023; Gao, 2023). These perturbations are minimal and preserve grammatical and semantic structure, ensuring that the modified step remains coherent but subtly altered. Additional implementation details are provided in Appendix C.

2.1 Context-based Average Treatment Effect

The Average Treatment Effect (ATE) (Rubin, 1974; Pearl, 2009) quantifies the causal effect of a binary intervention $\mathbf{X} \in \{0, 1\}$ on an outcome $\mathbf{Y}$ via Pearl's $\mathrm{do}(\cdot)$ operator:

$$\mathrm{ATE} = \mathrm{S}_1 - \mathrm{S}_0, \tag{1}$$

where $\mathrm{S}_{x \in \{0,1\}} := \Pr\big(\mathbf{Y}{=}1 \,\big|\, \mathrm{do}(\mathbf{X}{=}x)\big)$. To evaluate the causal contribution of a reasoning step s for LLMs' reasoning, we condition on a *context setting* $\mathbf{C} \in \{0, 1\}$ and define a *context-based ATE*:

$$\mathrm{ATE}(c) = \mathrm{S}_1(c) - \mathrm{S}_0(c), \tag{2}$$

$$\mathrm{S}_{x \in \{0,1\}}(c) := \Pr\big(\mathbf{Y}{=}1 \,\big|\, \mathbf{C}{=}c,\, \mathrm{do}(\mathbf{X}{=}x)\big), \tag{3}$$

where $c \in \{0, 1\}$ specifies the context regime and $\mathbf{X}$ toggles using the intact step s versus its perturbed counterpart s'. We consider two regimes: a) *Intact context* ($\mathbf{C}{=}1$): the original prefix $\mathcal{C}$ preceding s is kept as generated; b) *Perturbed context* ($\mathbf{C}{=}0$): all steps in $\mathcal{C}$ are minimally perturbed (e.g., by small numeric offsets), weakening associations between s and other steps so that the effect of s can be isolated.

Scoring the outcome. Let $y^* := f(q, \mathcal{C}^\star)$ be the model's reference answer obtained via *early-exit* on the full CoT, and let $f(q, \mathcal{C}, s)$ denote the early-exit prediction after a given $(\mathcal{C}, s)$. Instead of the binary $\mathbf{Y} = \mathbf{1}\{f(\cdot) = y^*\}$, we use the model's confidence for the event y^*: $\Pr(\mathbf{Y}{=}1 \mid \cdot) \equiv \Pr\big(y^* \mid q, \mathcal{C}, s\big)$, giving $\mathrm{S}_x(c)$ a probabilistic (confidence-based) interpretation.

Interpreting $\mathrm{ATE}_{\text{nec}}(1)$ vs. $\mathrm{ATE}_{\text{suf}}(0)$. Conditioning on $\mathbf{C}$ allows us to distinguish two complementary notions of causal relevance:

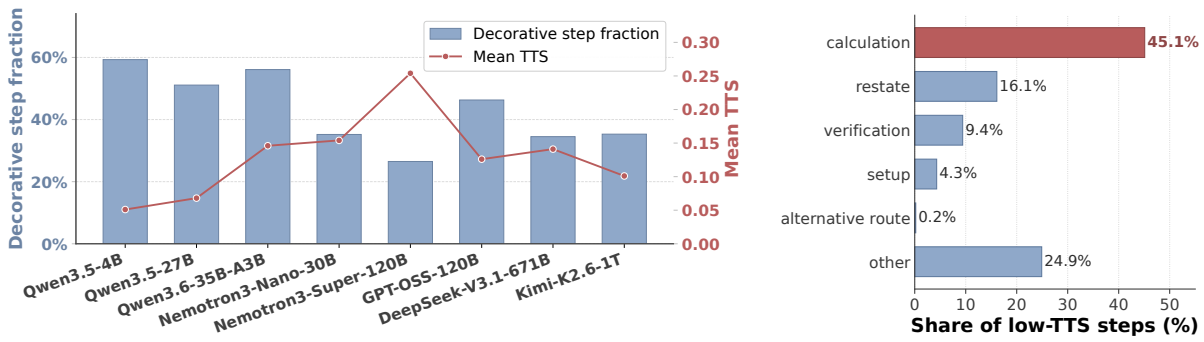
- **Necessity under intact context ($\mathrm{ATE}_{\text{nec}}(1)$).**

$$\mathrm{ATE}_{\text{nec}}(1) = \Pr(y^* \mid \mathbf{C}{=}1,\, \mathrm{do}(\mathbf{X}{=}1)) - \Pr(y^* \mid \mathbf{C}{=}1,\, \mathrm{do}(\mathbf{X}{=}0)).$$

  This tests whether s is *needed* given the full, supportive context $\mathcal{C}$. A low $\mathrm{ATE}_{\text{nec}}(1)$ indicates that removing s does not harm performance when other steps remain intact.

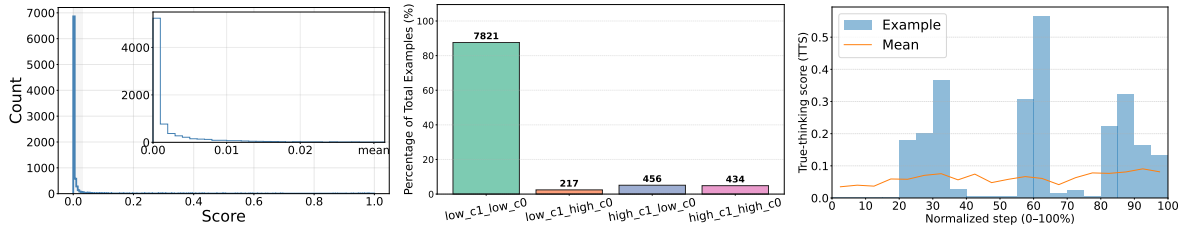
- **Sufficiency under perturbed context ($\mathrm{ATE}_{\text{suf}}(0)$).**

$$\mathrm{ATE}_{\text{suf}}(0) = \Pr(y^* \mid \mathbf{C}{=}0,\, \mathrm{do}(\mathbf{X}{=}1)) - \Pr(y^* \mid \mathbf{C}{=}0,\, \mathrm{do}(\mathbf{X}{=}0)).$$



(a) (b)

Figure 2: (a) Average fraction of steps with TTS ≤ 0.005 in CoT and average TTS across eight frontier models on MATH. Larger models contain fewer decorative reasoning steps, yet many steps still contribute minimally to the prediction, e.g., about 35% for Kimi-K2.6. Our results suggest that the latest models produce substantially performative and redundant CoT, which cannot be fully trusted as their true computation process. (b) Corresponding type breakdown of those low-TTS steps.



(a) (b) (c)

Figure 3: (a) The dataset-level distribution of the TTS score; (b) The distribution for $\mathrm{ATE}_{\mathrm{nec}}(1)$ and $\mathrm{ATE}_{\mathrm{suf}}(0)$ where *low* means ATE(·) is below mean and *high* means ATE(·) is above mean; (c) An example of TTS for steps at different percentile and the respective mean TTS.

This asks whether s can *on its own*—i.e., with weakened support from $\mathcal{C}$—drive the model toward y^*. A high $\mathrm{ATE}_{\mathrm{suf}}(0)$ suggests s is **sufficiently informative** to elicit the correct answer, capturing causal relevance even when s is not strictly necessary under the intact context.

Together, $\mathrm{ATE}_{\mathrm{nec}}(1)$ (necessity) and $\mathrm{ATE}_{\mathrm{suf}}(0)$ (sufficiency) provide a balanced view: A step can be causally meaningful by being necessary, sufficient, or both.

True-Thinking Score (TTS). Finally, the score is defined as

$$\mathrm{TTS}(s) = \tfrac{1}{2}\big(|\mathrm{ATE}_{\mathrm{nec}}(1)| + |\mathrm{ATE}_{\mathrm{suf}}(0)|\big). \quad (4)$$

A smaller $\mathrm{TTS}(s)$ indicates that the step has little causal influence on the model's prediction: perturbing or keeping it leads to almost the same result. Thus, that step is more likely to be *decorative* rather than *true thinking*. For each context setting c, we measure the unsigned $\mathrm{ATE}(c)$, $|\mathrm{ATE}(c)| = |S_1(c) - S_0(c)|$. The sign of $\mathrm{ATE}(c)$ reflects whether the step is helpful or harmful (e.g., the step is actually wrong) overall, but we are interested in *how much* the model truly thinks through the step in its internal computation, regardless of direction. Taking the absolute value thus captures the magnitude of a step's causal effect and provides a broader measure of its importance. TTS is a continuous metric, where lower values indicate that a step is more decorative.

3 Experimental Setup

Models. We conduct experiments on many different families of open-source reasoning models that have strong reasoning abilities and can generate long CoTs. See details in Appendix C.1.

Data. We evaluate on different reasoning benchmarks: AMC (American Mathematics Competitions), AIME (American Invitational Mathematics Examination) from 2020–2024, the MATH dataset (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021). For experiments in Section 5, we randomly sample 30% of cases as a heldout test set, 10 % of cases as validation set, and 60 % of cases as a training set.

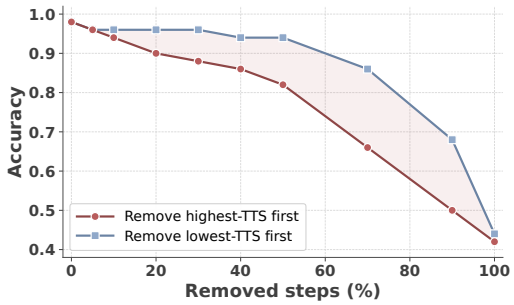
4 Evaluation of Step-wise Causality

As shown in Figure 2a, a significant fraction of CoT steps have TTS below 0.005, suggesting that many steps have minimal causal impact on the model's answer and may be decorative. Larger models generally achieve higher average TTS and contain fewer decorative steps than smaller models, such as 4B or 27B models. However, scaling does not linearly improve TTS: Kimi-K2.6 still has about 35% decorative steps, while Nemotron3-Super-120B achieves the highest TTS. These results suggest that a large fraction of CoT in current LLMs on practical reasoning questions may not be trusted as a faithful computation trace.

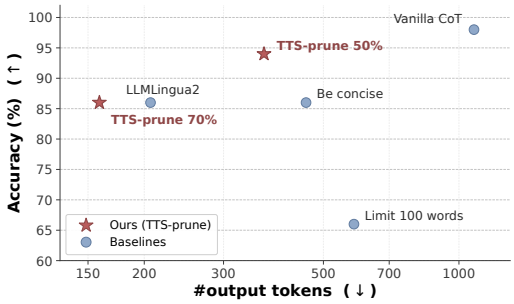
The distribution of TTS is long-tailed. As shown in Figure 3a for Deepseek-R1-Distill-Qwen-2.5-7B on AIME, we find most steps have low scores, while only a few have very high scores. For example, as shown in Figure 3a on the AIME dataset of Qwen-2.5, the mean is around 0.03. Only 6.4% of CoT steps achieve a TTS greater than 0.3, and merely 2.3% exceed 0.7. In Figure 3b, we observe disparities between $\mathrm{ATE}_{\mathrm{nec}}(1)$ and $\mathrm{ATE}_{\mathrm{suf}}(0)$ for the same step. This indicates that we need to consider both sufficiency and necessity to locate true-thinking steps. We confirm the same distribution pattern across different datasets (see Appendix D for mathematical reasoning and results on Commonsense reasoning in Appendix E).

Reasoning steps with high and low TTS are interleaved in a CoT. Figure 3c illustrates that steps with high TTS scores can appear at different positions, though later steps are on average more likely to be true-thinking with higher TTS. These results raise concerns about the reliability of monitoring LLMs by inspecting CoT (Baker et al., 2025), since individual steps may not always reflect the model's true computation and contribute to the model's final answer.

To further analyze low-TTS steps, we categorize them using the rules in Appendix A. As shown in Figure 2b, different types of reasoning steps can be decorative, with intermediate calculations accounting for most cases. Notably, self-verification steps can also be decorative (example shown in Figure 6). Overall, TTS quantifies the causal impact of reasoning traces and thus helps locate steps that may be redundant or may not reflect the model's true reasoning process.



(a)



(b)

Figure 4: (a) Pruning reasoning steps in Nemotron3-Super-120B on MATH. Pruning the lowest TTS steps preserves accuracy far longer than pruning the highest TTS steps. (b) TTS-prune sits on the efficiency–accuracy frontier among different baselines.

4.1 Pruning CoT with TTS

We further examine how pruning low- and high-TTS CoT steps affects reasoning performance in this section.

Pruning low-TTS steps retains the prompting performance. We rank all steps by TTS in ascending order, remove the lowest or highest $r\%$ of steps, and reprompt the model with the pruned CoT. As shown in Figure 4a, removing high-TTS steps quickly degrades performance, whereas pruning up to half of the lowest-TTS steps preserves most of the performance. Furthermore, we compare pruning the lowest-TTS steps with other methods for producing shorter CoT: (1) *Be concise* (Renze and Guven, 2024), which directly prompts the model to be concise; (2) *Limit the answer length to 100 words* (Han et al., 2025), which prompts LLMs to control the generation budget. This can shorten CoT, though generations may still exceed the token limit; (3) LLMLingua-2, a trained LM that labels semantically important tokens to remove the unimportant ones (Pan et al., 2024; Xia et al., 2025). As shown in Figure 4b, pruning 50% of the steps from the lowest TTS provides a good balance between token length and reasoning performance compared with these methods. Pruning 70% further shortens

the CoT, but sacrifices performance a bit.

Variant	MATH		GSM8K	
	Acc. ↑	#Token ↓	Acc. ↑	#Token ↓
Base	88	732	76	349
TokenSkip (Xia et al., 2025)	84	686	76	308
ST-ICL (Munkhbat et al., 2025)	79	496	79	324
ST-TTS-prune (ours)	**88**	**488**	**87**	**279**

Table 1: Performance of different training methods to improve reasoning efficiency.

Further self-training can improve reasoning efficiency. We further demonstrate that self-training (ST) (Xie et al., 2020) on TTS-pruned CoT (pruning rate=50%) can reduce the model's CoT length, while preserving the performance. We compare against TokenSkip (Xia et al., 2025), which trains on CoT pruned by LLMLingua-2, and ST-ICL (Munkhbat et al., 2025), which self-trains on concise CoT generated through in-context learning. We use LoRA (Hu et al., 2022) (rank=32) to fine-tune Nemotron3-nano-30B on 300 pruned CoT examples from MATH for five epochs, and evaluate on held-out MATH and GSM8K.

As shown in Table 1, ST-TTS-prune produces the fewest reasoning tokens while maintaining the best reasoning performance among the compared methods. On MATH, it reduces reasoning tokens by 66% relative to the base model, and on GSM8K it also improves performance. In contrast, TokenSkip only modestly reduces reasoning length, while ST-ICL substantially degrades performance. Neither baseline appears to generalize well to GSM8K in terms of improving reasoning efficiency.

Overall, TTS provides a useful signal for pruning CoT that can be used in student-model training (Guo et al., 2025) or self-training for more efficient reasoning (Munkhbat et al., 2025). It may also be explored as a reward signal in reinforcement learning (Guo et al., 2025; Lambert et al., 2024) to reduce steps with low causal impact and improve CoT efficiency and trustworthiness, which we leave to future work.

5 Mediating Reasoning via Steering

In this section, we further provide mechanism insights on decorative thinking in CoT through steering. We empirically reveal that, in LLMs, latent-space steering with the TrueThinking direction identified by TTS can mediate whether to use a reasoning step in CoT when computing the results. Steering the hidden states of a step along TrueThinking direction can force LLMs to engage with that step to compute its answer and may increase the attention to that step as well and vice versa.

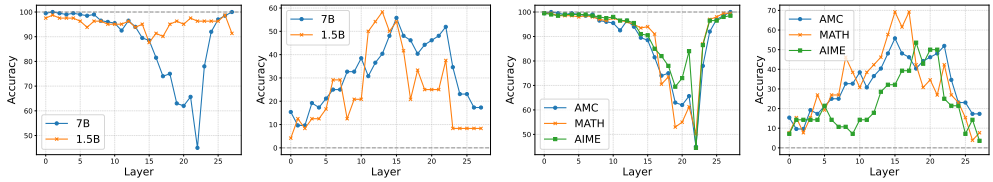
Extracting the steering direction with TTS. We first detail how to extract the steering direction. We focus on the residual stream activation $h^l(s_t)$ of the last token position t for a step s at a layer l. At a layer l, we collect the hidden states of the most representative steps s_{TT} (where $\text{TTS}(s_{\text{TT}}) \geq$ threshold α) and steps s_{DT} (where $\text{TTS}(s_{\text{DT}}) \leq$ threshold β). Following the difference-in-means approach (Marks and Tegmark, 2023; Arditi et al., 2024; Zhao et al., 2025), we compute the direction as the mean shift from $\mu^l_{\text{TT}} = \text{mean}(h^l(s_{\text{TT}}))$ to $\mu^l_{\text{DT}} = \text{mean}(h^l(s_{\text{DT}}))$ in the latent space, i.e., $v^l_{\text{TrueThinking}} = \mu^l_{\text{TT}} - \mu^l_{\text{DT}}$. For steering at test time, we modify the residual stream for the hidden state of a test step in the example by using activation addition at a single layer l, i.e., $\bar{h}^l = h^l + v^l_{\text{TrueThinking}}$ to all tokens in the step. Detailed implementations are in Appendix C.2.

Specifically, we have two steering cases, **(1) Engagement Test**: We consider cases where $f(q, \mathcal{C}) = y_{GT}$ and $f(q, \mathcal{C}, s') = y_{GT}$. Namely, the model still obtains the ground-truth answer y_{GT} despite the perturbed step s'. If we apply a steering direction v^l to the hidden state of s', and the model's correct answer flips to an incorrect one ($f^{+v^l}(q, \mathcal{C}, s') \neq y_{GT}$), this indicates that the steering has forced the model to reason over s', thus following the errors injected into s'. **(2) Disengagement Test**: On the other hand, we consider the reverse cases where the model first predicts the correct answer, i.e., $f(q, \mathcal{C}) = y_{GT}$, but including a perturbed step s' causes it to fail: $f(q, \mathcal{C}, s') \neq y_{GT}$. If applying a steering direction $-v^l$ to s' flips the wrong answer to the correct answer ($f^{-v^l}(q, \mathcal{C}, s') = y_{GT}$), then the intervention has made the model disregard the step s'.

As comparison baselines, we consider three approaches for layer-wise intervention apart from the TrueThinking direction. (1) DropStep direction: Motivated by the prior work (Tanneru et al., 2024; Lanham et al., 2023; Bogdan et al., 2025; Wang et al., 2025) on faithfulness evaluation, we compare prediction confidence before and after dropping step s, i.e., $\text{P}(y_{GT}|q, \mathcal{C}, o) - \text{P}(y_{GT}|q, \mathcal{C})$ (equivalent to $\text{ATE}_{\text{nec}}(1)$ in our framework). The steps of the highest and lowest scores are then used to extract a steering direction following the same

Dataset / Method	Attention scaling		Random vector		DropStep direction		TrueThinking direction	
	ET	DT	ET	DT	ET	DT	ET	DT
AMC								
Qwen-2.5	6.2	25.0	4.0	26.9	1.5	28.6	**55.0**	**55.7**
Llama-3.1	**24.8**	20.5	3.5	20.6	10.5	32.4	17.6	**35.3**
Nemotron	5.1	27.2	4.5	45.5	9.0	45.4	**35.7**	**54.5**
MATH								
Qwen-2.5	10.0	23.9	2.0	30.2	2.5	17.7	**49.8**	**69.2**
Llama-3.1	7.5	35.4	5.0	47.9	11.0	52.1	**14.0**	**54.2**
Nemotron	21.7	42.7	21.5	44.6	6.5	45.1	**59.5**	**56.3**
AIME								
Qwen-2.5	9.3	25.0	1.5	21.4	1.5	14.3	**55.5**	**53.6**
Llama-3.1	6.3	35.2	2.5	29.4	5.0	41.1	**38.0**	**47.1**
Nemotron	12.0	70.6	6.5	76.5	4.5	79.5	**39.0**	**91.2**

Table 2: Top-1 flip rate among all layers (%) ↑ in the Engagement Test (ET) and the Disengagement Test (DT). We use flip rate as the metric, measuring how often steering changes the model's initial prediction. AMC dataset is in-domain evaluation where TrueThinking directions are extracted, while the other two datasets are for out-of-domain evaluation.



(a) Engagement Test (b) Disengagement Test (c) Engagement Test (d) Disengagement Test

Figure 5: Layer-wise results of steering with the TrueThinking vector. In the Engagement Test, stronger intervention is reflected by lower accuracy (more right→wrong flips); In the Disengagement Test, by higher accuracy (more wrong→right flips). Figures (a–b): layer-wise results on AMC for DeepSeek-R1-Distill-Qwen-7B and its 1.5B variant under the Engagement Test and the Disengagement Test. Figures (c–d): cross-domain results, where the TrueThinking direction is extracted on AMC and applied to MATH and AIME.

method as the TrueThinking direction. (2) Random steering vector: we generate a random vector with the same dimensionality and norm as the TrueThinking direction to test whether our identified direction encodes meaningful information. (3) Attention scaling: we scale the attention weights of the tokens of step s at a layer during inference time, with scale=100 encouraging the model to think through the step and scale=0 suppressing it.

5.1 Whether to use a reasoning step can be steerable

As shown in Table 2, steering with the (reverse) TrueThinking direction reliably flips predictions in both tests. In the Disengagement Test, it effectively prevents the model from using the perturbed step s', with effects far stronger than those of random vectors. This shows that suppression of step use with the reverse TrueThinking direction in the Disengagement Test arises from a meaningful signal rather than noise. Additionally, the TrueThinking direction derived from our TTS can produce stronger and more consistent intervention effects than the steering directions extracted using DropStep.

On the other hand, our experiments across datasets show that the latent signal controlling whether a step engages in reasoning is universal. As seen in Table 2, the TrueThinking direction extracted on AMC generalizes well to other datasets across all models, indicating a model-internal mechanism of thinking rather than a dataset-specific artifact. In the Qwen-2.5 model, layer 15-22 consistently yield the strongest intervention performance across all three datasets (Figure 5c–5d), suggesting these intermediate layers may concentrate latent reasoning. We also compare models with different sizes in the same model family. Similar patterns hold for the much smaller Qwen-2.5-1.5B model (Figure 5a and Figure 5b) and larger 14B model (Figure 11 in Appendix F). Steering along the TrueThinking direction (to in-

duce the step use in LLMs' internal reasoning) is weaker for smaller models, whereas the results in the Disengagement Test are comparable.

Steering with the TrueThinking direction mediates LLMs' attention. We find that steering with the TrueThinking direction can reallocate attention among tokens. In Disengagement Test, steering in the reverse TrueThinking direction reduces attention to the perturbed tokens (see examples in Figures 8c and 8d of the Appendix), causing the model to largely disregard them. Similarly, steering along the TrueThinking direction increases attention to the step (see examples in Figure 8a and Figure 8b of the Appendix). Overall, when steering along the TrueThinking direction, the attention of the steered tokens will increase on average by 193% and steering along the reverse direction will decrease the attention by 32.3% across the layers.

Steering LLMs for true self-verification. We define decorative-thinking self-verification $\mathcal{S}_{\text{DT-self-verify}} = (s_1, s_2, \ldots, s_n)$ where $\text{TTS}(s_i)$ is smaller than 0.005. We focus on a setting in which the model fails to follow correct steps in self-verification $\mathcal{S}_{\text{DT-self-verify}}$ and is instead misled by incorrect previous steps $\mathcal{C}'$, resulting in a wrong prediction, i.e., $f(q, \mathcal{C}', \mathcal{S}_{\text{DT-self-verify}}) \neq y_{GT}$. We apply the TrueThinking direction to the tokens in $\mathcal{S}_{\text{DT-self-verify}}$. We find that this can at best recover around 51% of the cases where the models do not follow their correct self-verification steps in CoT and give wrong answers (layer-wise results shown in Figure 7 of Appendix). Remarkably, the layer with the strongest steering effect aligns with the layer identified in Figure 5c–5d, suggesting that certain layers may play a special role in determining whether reasoning steps are engaged in the model's internal computation.

6 Related Work

Internal thinking in LLMs' latent space. Apart from explicit CoT, LLMs "think" internally across their layers. They can directly answer reasoning problems, sometimes even matching the performance of CoT-based prompting (Ma et al., 2025). Prior work suggests that LLMs may solve certain tasks through internal *circuits* (Yang et al., 2024; Marks et al., 2024; Prakash et al., 2025). The *implicit* reasoning capabilities of LLMs may bypass using explicit CoTs (Deng et al., 2023; Hao et al., 2024; Pfau et al., 2024; Goyal et al., 2023). Such internal reasoning of LLMs questions how much the model truly uses each step in their CoT, which we study using a causal framework.

The unfaithfulness of LLMs' reasoning traces. Prior work has raised broad concerns about the faithfulness of reasoning traces. Models may rely on hidden hints or spurious biases without acknowledging them in CoT (Chen et al., 2025b; Arcuschin et al., 2025; Chua and Evans, 2025; Turpin et al., 2023), and generated rationales may be insensitive to perturbations or unnecessary for reaching the final answer (Gao, 2023; Lanham et al., 2023; Tanneru et al., 2024; Wang et al., 2025). While these studies provide important evidence that CoT can be unfaithful or post-hoc, they mostly evaluate CoT holistically, leaving the causal role of individual steps underexplored.

Steering LLMs. Steering directions applied to intermediate hidden states have been found to mediate LLMs' behaviors or perception in many aspects (Von Rütte et al., 2024; Turner et al., 2023; Tigges et al., 2023; Li et al., 2023; Marks and Tegmark, 2023; Zhao et al., 2025). In terms of reasoning, past works have found steering vectors that can control the strength of reasoning, e.g., longer or shorter CoT (Tang et al., 2025; Sun et al., 2025; Chen et al., 2025a; Sheng et al., 2025) or different reasoning styles in CoT (Venhoff et al., 2025).

7 Conclusions

We propose True-Thinking Score to analyze step-wise causality in CoT and show that frontier LLMs contain many decorative steps that make minimal causal contribution to the model's answer on common reasoning benchmarks. We reveal that *true-thinking* and *decorative-thinking* steps are often interleaved: only a small subset causally influences the final prediction, while many steps merely create the appearance of reasoning. We further show that TTS can prune CoTs into concise rationales for self-training, substantially improving reasoning efficiency without sacrificing performance. Mechanistically, we demonstrate that whether a reasoning step contributes to the model's computation can be mediated by a TrueThinking direction, suggesting that step-level use is reflected in internal representations. Broadly, our findings raise concerns about both the efficiency of CoT reasoning of current LLMs and the reliability of using CoT to interpret LLMs' internal thought.

8 Limitations

Our causal evaluation framework is inherently approximate and especially focuses on math reasoning that has strong logic. It is greedy in nature and may not capture all possible causal pathways, nor does it aim to reconstruct a complete causal graph of reasoning steps. Thus, it should be viewed as a probe that highlights representative *true-thinking* and *decorative-thinking* steps rather than a definitive oracle of internal reasoning. However, we can still leverage TTS to prune CoT effectively. In addition, the TrueThinking direction we extract may not be optimal. We regard our findings as an existence proof that whether to engage with a reasoning step can be mediated by steering directions, and we leave the development of more effective directions and a deeper understanding of their geometry to future work.

References

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, and 1 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.

Iván Arcuschin, Jett Janiak, Robert Krzyzanowski, Senthooran Rajamanoharan, Neel Nanda, and Arthur Conmy. 2025. Chain-of-thought reasoning in the wild is not always faithful. *arXiv preprint arXiv:2503.08679*.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. 2024. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems*, 37:136037–136083.

Bowen Baker, Joost Huizinga, Leo Gao, Zehao Dou, Melody Y Guan, Aleksander Madry, Wojciech Zaremba, Jakub Pachocki, and David Farhi. 2025. Monitoring reasoning models for misbehavior and the risks of promoting obfuscation. *arXiv preprint arXiv:2503.11926*.

Aaron Blakeman, Aaron Grattafiori, Aarti Basant, Abhibha Gupta, Abhinav Khattar, Adi Renduchintala, Aditya Vavre, Akanksha Shukla, Akhiad Bercovich, Aleksander Ficek, and 1 others. 2025. Nvidia nemotron 3: Efficient and open intelligence. *arXiv preprint arXiv:2512.20856*.

Paul C Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. 2025. Thought anchors: Which llm reasoning steps matter? *arXiv preprint arXiv:2506.19143*.

Runjin Chen, Zhenyu Zhang, Junyuan Hong, Souvik Kundu, and Zhangyang Wang. 2025a. Seal: Steerable reasoning calibration of large language models for free. *arXiv preprint arXiv:2504.07986*.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, and 1 others. 2025b. Reasoning models don't always say what they think. *arXiv preprint arXiv:2505.05410*.

James Chua and Owain Evans. 2025. Are deepseek r1 and other reasoning models more faithful? *arXiv preprint arXiv:2501.08156*.

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*.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. 2023. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. 2025. Deep think with confidence. *arXiv preprint arXiv:2508.15260*.

Leo Gao. 2023. Shapley value attribution in chain of thought. *URL https://www. lesswrong. com/posts/FX5JmftqL2j6K8dn4/shapley-value-attribution-in-chain-of-thought*.

Olga Golovneva, Moya Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2022. Roscoe: A suite of metrics for scoring step-by-step reasoning. *arXiv preprint arXiv:2212.07919*.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. 2023. Think before you speak: Training language models with pause tokens. *arXiv preprint arXiv:2310.02226*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

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 E. Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*.

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. *arXiv preprint arXiv:2103.03874*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2025. C3ot: Generating shorter chain-of-thought without compromising effectiveness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24312–24320.

Tomek Korbak, Mikita Balesni, Elizabeth Barnes, Yoshua Bengio, Joe Benton, Joseph Bloom, Mark Chen, Alan Cooney, Allan Dafoe, Anca Dragan, and 1 others. 2025. Chain of thought monitorability: A new and fragile opportunity for ai safety. *arXiv preprint arXiv:2507.11473*.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, and 1 others. 2023. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36:41451–41530.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. 2025. Reasoning models can be effective without thinking. *arXiv preprint arXiv:2504.09858*.

Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. 2024. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *arXiv preprint arXiv:2403.19647*.

Samuel Marks and Max Tegmark. 2023. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *arXiv preprint arXiv:2310.06824*.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*.

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.

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.

Judea Pearl. 2009. *Causality*. Cambridge university press.

Jacob Pfau, William Merrill, and Samuel R Bowman. 2024. Let's think dot by dot: Hidden computation in transformer language models. *arXiv preprint arXiv:2404.15758*.

Nikhil Prakash, Natalie Shapira, Arnab Sen Sharma, Christoph Riedl, Yonatan Belinkov, Tamar Rott Shaham, David Bau, and Atticus Geiger. 2025. Language models use lookbacks to track beliefs. *arXiv preprint arXiv:2505.14685*.

Matthew Renze and Erhan Guven. 2024. The benefits of a concise chain of thought on problem-solving in large language models. *arXiv preprint arXiv:2401.05618*.

Donald B Rubin. 1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5):688.

Leheng Sheng, An Zhang, Zijian Wu, Weixiang Zhao, Changshuo Shen, Yi Zhang, Xiang Wang, and Tat-Seng Chua. 2025. On reasoning strength planning in large reasoning models. *arXiv preprint arXiv:2506.08390*.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Jiwon Song, Dongwon Jo, Yulhwa Kim, and Jae-Joon Kim. 2025. Reasoning path compression: Compressing generation trajectories for efficient llm reasoning. *arXiv preprint arXiv:2505.13866*.

Chung-En Sun, Ge Yan, and Tsui-Wei Weng. 2025. Thinkedit: Interpretable weight editing to mitigate overly short thinking in reasoning models. *arXiv preprint arXiv:2503.22048*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Xinyu Tang, Xiaolei Wang, Zhihao Lv, Yingqian Min, Wayne Xin Zhao, Binbin Hu, Ziqi Liu, and Zhiqiang Zhang. 2025. Unlocking general long chain-of-thought reasoning capabilities of large language models via representation engineering. *arXiv preprint arXiv:2503.11314*. ACL 2025.

Sree Harsha Tanneru, Dan Ley, Chirag Agarwal, and Himabindu Lakkaraju. 2024. On the hardness of faithful chain-of-thought reasoning in large language models. *arXiv preprint arXiv:2406.10625*.

Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. 2023. Linear representations of sentiment in large language models. *arXiv preprint arXiv:2310.15154*.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. 2025. Understanding reasoning in thinking language models via steering vectors. *arXiv preprint arXiv:2506.18167*.

Dimitri Von Rütte, Sotiris Anagnostidis, Gregor Bachmann, and Thomas Hofmann. 2024. A language model's guide through latent space. *arXiv preprint arXiv:2402.14433*.

Zezhong Wang, Xingshan Zeng, Weiwen Liu, Yufei Wang, Liangyou Li, Yasheng Wang, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. 2025. Chain-of-probe: Examining the necessity and accuracy of CoT step-by-step. In *Findings of the Association for Computational Linguistics: NAACL 2025*.

Tom Wollschläger, Jannes Elstner, Simon Geisler, Vincent Cohen-Addad, Stephan Günnemann, and Johannes Gasteiger. 2025. The geometry of refusal in large language models: Concept cones and representational independence. In *Forty-second International Conference on Machine Learning*.

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.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10687–10698.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Zheng Lin, Li Cao, and Weiping Wang. 2025b. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. 2024. Do large language models latently perform multi-hop reasoning? *arXiv preprint arXiv:2402.16837*.

Evelyn Yee, Alice Li, Chenyu Tang, Yeon Ho Jung, Ramamohan Paturi, and Leon Bergen. 2024. Dissociation of faithful and unfaithful reasoning in llms. *arXiv preprint arXiv:2405.15092*.

Xiangning Yu, Zhuohan Wang, Linyi Yang, Haoxuan Li, Anjie Liu, Xiao Xue, Jun Wang, and Mengyue Yang. 2025. Causal sufficiency and necessity improves chain-of-thought reasoning. *arXiv preprint arXiv:2506.09853*.

Jiachen Zhao, Jing Huang, Zhengxuan Wu, David Bau, and Weiyan Shi. 2025. Llms encode harmfulness and refusal separately. *arXiv preprint arXiv:2507.11878*.

Ziqian Zhong, Aditi Raghunathan, and Nicholas Carlini. 2025. Impossiblebench: Measuring llms' propensity of exploiting test cases. *arXiv preprint arXiv:2510.20270*.

Problem. Simplify $\frac{\sec x}{\sin x} - \frac{\sin x}{\cos x}$.

Preceding context:

```
...Express everything in terms of
sin and cos.  Common denominator:
sin x/ cos x = sin^2 x/(sin x cos x).
Difference: (1 − sin^2 x)/(sin x cos x) =
cos^2 x/(sin x cos x) = cos x/ sin x = cot x.
```

Original step:

```
Check: sec x/ sin x = 1/(cos x sin x)
sin x/ cos x = sin^2 x/(cos x sin x)
(1 − sin^2 x)/(cos x sin x) = cos^2 x/(cos x sin x)
```

Perturbed step:

```
Check: sec x/ sin x = 4/(cos x sin x)
sin x/ cos x = (sin^−1 x)/(cos x sin x) (−2 −
sin^5 x)/(cos x sin x) = cos^1 x/(cos x sin x)
```

Figure 6: Example decorative verification step of Nemotron3-Nano. Perturbing the step with intact and perturbed context shifts confidence of model's predicted answer ($\cot x$) by only 6.4e-12 and 4.5e-14 respectively.

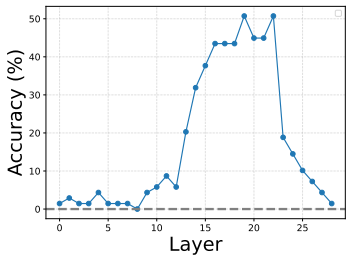


Figure 7: Layer-wise results after steering the model to truly self-verify their reasoning steps. Initially, the test accuracy is zero without steering, as the model ignores its self-verification steps in CoT and outputs wrong answers.

A Taxonomy of Low-TTS Reasoning Steps

To better understand which reasoning steps receive low TTS, we categorize low-TTS steps into a small set of functional types. This taxonomy is intended as an interpretability tool rather than a learned classifier. Each step is assigned to exactly one category using a priority-ordered rule set based on lexical and structural cues, so that broad classes such as calculation do not absorb more specific functions such as verification or restatement.

Verification. Steps that explicitly check or validate a previous result, with cues such as "check," "verify," "substitute," or "sanity check."

Restatement. Steps that restate an intermediate result/ condition or summarize a derivation, often with cues such as "therefore," "thus".

Calculation. Steps that contain explicit numerical or algebraic manipulation, identified by mathematical operators, equations, fractions, or digits. This class is particularly important: low-TTS calculations show that even symbolic-looking computation can be causally inert. In many cases, perturbing numbers in these steps leaves the final answer unchanged, suggesting that the calculation is redundant or post-hoc.

Setup. Steps that introduce notation, variables, or representations, with cues such as "let," "define," "denote," "set," or "represent." Low-TTS setup steps often correspond to conventional problem translation: they may make the solution easier to read, but the model may not rely on them to compute the answer.

Alternative route. Steps that introduce an optional solution path, with cues such as "alternative," "another way," or "could also."

Other. Steps that do not match the above patterns are assigned to an "other" category. This residual bucket captures heterogeneous cases and is not interpreted as a coherent reasoning type.

B More related work

CoT pruning. Our work is also related to CoT compression (Xia et al., 2025; Song et al., 2025; Kang et al., 2025; Yu et al., 2025) and important steps analysis (Golovneva et al., 2022). However, those works are different as they focus on the semantics of verbalized steps for analysis and do not look into the underlying causality. For example, Xia et al. (2025) prune long CoTs into shorter sentences with equivalent meanings for LLMs without degrading performance. But that does not mean pruned steps do not causally contribute to the model's answer. Our work may suggest similar results that there exist shorter alternative CoTs leading the model to get the same answers, but they do not solve whether a step truly engages with the model's computation, and the internal mechanism of LLMs' thinking process is still unclear. Yu et al. (2025) reframe CoT pruning into the Probability of Necessity and Sufficiency (PNS) framework (Pearl, 2009), but it focuses on estimating the necessity of each step using counterfactual intervention in a similar way to prior works (Lanham et al., 2023; Yee et al., 2024; Bogdan et al., 2025). Although they are framed into PNS, their measurement of sufficiency and necessity is restricted to contexts where the prior steps remain unchanged, which corresponds to $\mathrm{ATE}_{\text{nec}}(1)$ in our framework.

C Implementations

Perturbing reasoning steps. We treat sentences as distinct reasoning steps, as prior work has shown that each sentence can serve a different function within a reasoning trace (Bogdan et al., 2025). We follow prior work (Lanham et al., 2023; Gao, 2023) and add small random offsets (chosen from $[-3, -2, -1, 1, 2, 3]$) to the numbers in a reasoning step. This keeps the perturbation minimal so that the step remains largely unchanged in token length, wording, and underlying logic. We can therefore reasonably attribute any confidence changes caused by the perturbation to the model's treatment of the original step. For steps that do not contain numerical values, we also follow prior work (Bogdan et al., 2025; Tanneru et al., 2024; Wang et al., 2025) by dropping them as a perturbation to measure the influence of those sentences. For perturbing context steps, we only change numerical values.

C.1 Models

We evaluate on different open-weight LLMs spanning 1.5B to 1T parameters. For Qwen-2.5-7B and Llama-3.1-8B, we use the versions fine-tuned on samples generated by DeepSeek-R1 (Guo et al., 2025), i.e., DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Llama-8B, along with Nemotron-1.5B; these models generate very long CoTs at test time. From the latest Qwen3 family (Yang et al., 2025a), we include Qwen3.5-4B, Qwen3.5-27B, and Qwen3.6-35B-A3B. We additionally evaluate Nemotron3-Nano-30B and Nemotron3-Super-120B (Blakeman et al., 2025), GPT-OSS-120B (Agarwal et al., 2025), DeepSeek-V3.1-671B (Liu et al., 2024), and Kimi-K2.6-1T[1]. This set covers dense and Mixture-of-Experts architectures, transformer and hybrid Mamba-Transformer designs, and both reasoning-tuned and general-purpose post-training recipes.

C.2 Steering

To compute the TrueThinking direction, for all tested models we set the threshold α=0.9 for s_{TT}, while β=0 for s_{DT}, which means perturbing those steps s_{DT} does not change the model's confidence at all. We emphasize that thresholding is used only to select the most representative steps for extracting the steering direction, while TTS is a continuous measure. The ablation study for thresholding is in Appendix G.1.

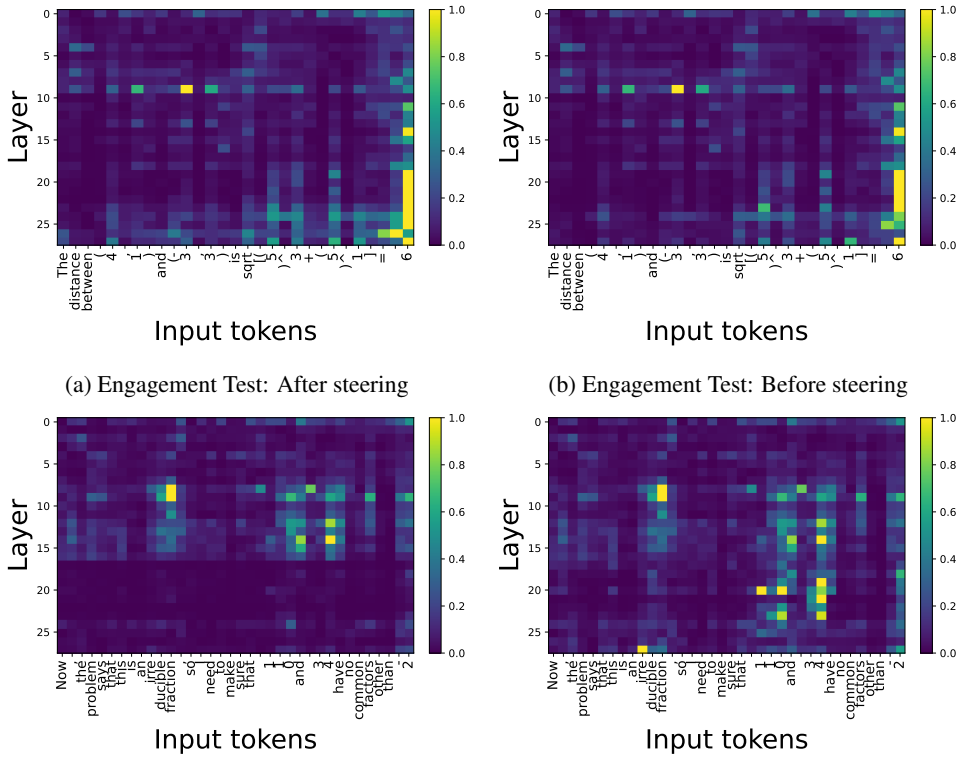
D More Experimental Results of TTS

As shown in Figure 9, we observe a similar long-tail distribution of TTS across different datasets. Steps with larger TTS are sparse, while the most of steps have relatively low TTS. Additionally, as shown in Figure 9b and Figure 9d, on different datasets, we still observe that there exist ample examples where $\mathrm{ATE}_{\text{nec}}(1)$ and $\mathrm{ATE}_{\text{suf}}(0)$ of the same step are very different.

E Results on Commonsense Reasoning

We apply TTS and the TrueThinking direction to commonsense reasoning to test the generalizability of our framework to more generic reasoning cases. We use CommonsenseQA (Talmor et al., 2019) as our test data. Since commonsense reasoning is predominantly non-numeric, standard intervention schemes that rely on perturbing numerical values

[1] https://huggingface.co/moonshotai/Kimi-K2.6



(a) Engagement Test: After steering

(b) Engagement Test: Before steering

(c) Disengagement Test: After steering

(d) Disengagement Test: Before steering

Figure 8: Normalized attention scores of the step in the Engagement Test and the Disengagement Test before and after steering at Layer 17. (a–b) Applying the TrueThinking direction to a step increases the model's attention to it. (c–d) Applying the reverse TrueThinking direction decreases the model's attention.

Question	Where would you find magazines alongside many other printed works?
Options	**A** doctor **B** ✓ bookstore **C** market **D** train station **E** mortuary
Original rationale	...bookstores are designed to sell both books and periodicals...
Perturbed rationale	...train stations are designed to sell both books and periodicals...

Table 3: Example multiple-choice question with original vs. perturbed rationale.

are not suitable in this setting. To apply interventions on such content, we prompt a separate large language model (GPT-5) to produce subtle, semantically coherent distortions of the initial reasoning step. This makes perturbation applicable to different reasoning scenarios. An example perturbed rationale is shown in Table 3.

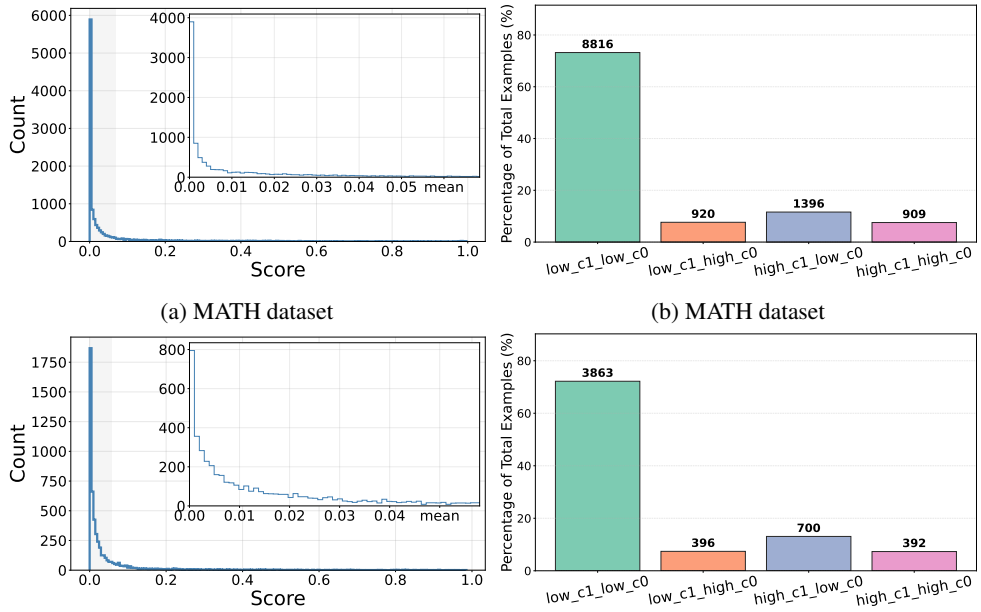
As shown in Figure 10a and Figure 10b, the distribution of TTS on Commonsense reasoning is long-tailed, which is similar to the patterns observed on mathematical reasoning.

Furthermore, we conduct the steering tests (explained in Section 5) on the examples of Commonsense QA. Specifically, we use the same TrueThinking direction extracted from AMC dataset for steering (the same setting as Table 2). The steering results are shown in Figure 10c. Noticeably, the TrueThinking direction can still mediate the reasoning behaviors on CommonsenseQA when applied to the middle layers. This strongly indicates the generalizability of TrueThinking direction and suggests that LLMs encode a specific variable for true-thinking behaviors.

F Steering Results on Larger Models

Beyond the 1.5B and 7B models in Figure 5a and Figure 5b, we further demonstrate that the TrueThinking direction generalizes to a larger 14B model. As shown in Figure 11, we can similarly modulate the 14B model's reasoning behavior, controlling whether it engages or disengages with reasoning steps by steering the hidden states in the middle layers along or against TrueThinking direction.

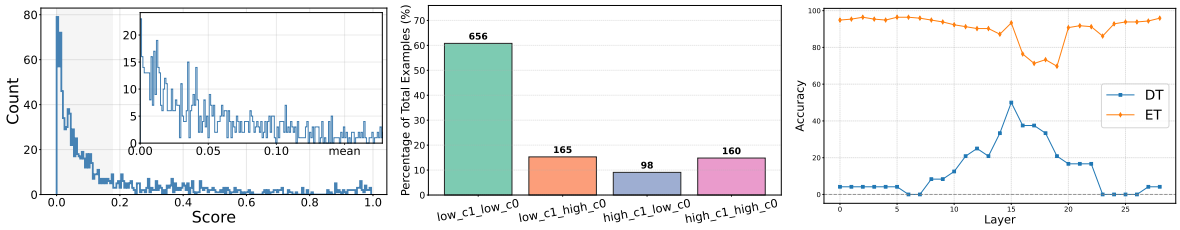
For instance, in the Qwen-2.5 model, layer 15-22



(a) MATH dataset (b) MATH dataset

(c) AMC dataset (d) AMC dataset

Figure 9: Distribution of TTS on different datasets.



(a) (b) (c)

Figure 10: Results for Commonsense reasoning. (a) The dataset-level distribution of the TTS score; (b) The distribution for $\mathrm{ATE}_{\mathrm{nec}}(1)$ and $\mathrm{ATE}_{\mathrm{suf}}(0)$ where *low* means $\mathrm{ATE}(\cdot)$ is below mean and *high* means $\mathrm{ATE}(\cdot)$ is above mean; (c) Steering results on the Engagement Test and the Disengagement Test.

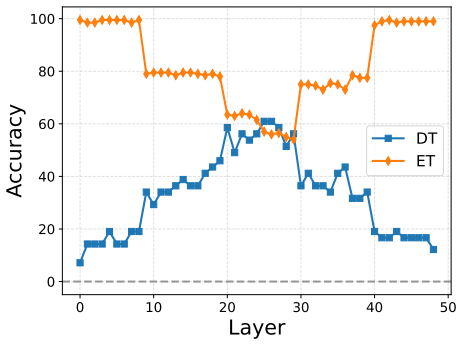


Figure 11: Results of Engagement Test and Disengagement Test on Qwen-14B.

consistently yield the strongest intervention performance across all three datasets (Figure 5c–5d), suggesting these intermediate layers concentrate latent reasoning. We also compare models with different sizes in the same model family. Similar patterns hold for the much smaller Qwen-2.5-1.5B model (Figure 5a and Figure 5b) and larger 14B model (Figure 11 in Appendix F). Steering along the TrueThinking direction (to induce the step use in LLMs' internal reasoning) is weaker for smaller models, whereas the results in the Disengagement Test are comparable.

G Ablation study

G.1 Threshold of TTS for extracting steering directions

In this section, we ablate the threshold for selecting true-thinking steps when extracting the TrueThinking vector. Our goal is to better understand the

scale of TTS, that is, how low a score may already indicate decorative thinking and how high a score reflects true thinking. We use steering performance as an indirect probe of how TTS correlates with the internal engagement of steps in reasoning.

When extracting steering directions with difference-in-means, the steps with zero TTS are treated as decorative-thinking steps (s_{DT}), while we use steps from different ranges of TTS as true-thinking steps (s_{TT}). As shown in Figure 12, when TrueThinking directions are computed from steps drawn from increasingly higher TTS ranges, the steering effect grows roughly linearly. In contrast, using steps with TTS below 0.03 yields negligible steering, suggesting these steps are internally treated as decorative, similarly to those with zero TTS. Because the TrueThinking directions are computed as the difference in mean hidden states between true and decorative steps (Eq. **??**), negligible steering effects mean the direction fails to capture the meaningful distinction between decorative and true thinking.

Overall, this analysis reveals an implicit decision boundary in the model's internal space: while the distribution of TTS is very long-tailed and high-TTS steps are rare, larger TTS indeed corresponds to genuinely influential reasoning. We leave further in-depth study of the geometry (Wollschläger et al., 2025) of hidden states and TrueThinking directions in the latent space as future work.

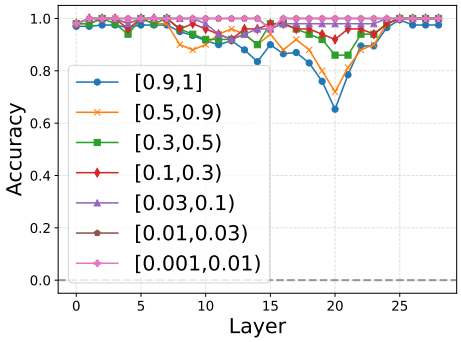


Figure 12: Test results of Nemotron on the Engagement Test where TrueThinking directions are extracted between examples with zero TTS (as decorative-thinking steps s_{DT}) and examples of different ranges of TTS (as true-thinking steps s_{TT}), and the lower accuracy means stronger steering effects.

G.2 Averaging over ATE(c) for TTS

We compare using only ATE(1) in TTS with the complete TTS to identify true-thinking steps. This slightly differs from the DropStep method in Table 4, which is adopted by past works (Tanneru et al., 2024; Lanham et al., 2023; Wang et al., 2025; Bogdan et al., 2025). DropStep defines ATE(1) with step removal as the perturbation, i.e., $\mathrm{ATE}(1)^{\mathrm{drop}} = \mathrm{P}(y^* \mid \mathcal{C}, s) - \mathrm{P}(y^* \mid \mathcal{C})$. In contrast, here $\mathrm{ATE}(1) = \mathrm{P}(y^* \mid \mathcal{C}, s) - \mathrm{P}(y^* \mid \mathcal{C}, s')$ using numerical perturbation by changing the numbers in step s. However, as shown in Table 4, incorporating ATE(0) is still necessary, as the resulting TrueThinking direction achieves stronger intervention performance.

Table 4: Results on MATH for steering with directions identified from true-thinking steps based on ATE(1) and complete TTS.

	$\lvert\mathrm{ATE}_{\mathrm{nec}}(1)\rvert$ only		$\mathrm{mean}(\lvert\mathrm{ATE}_{\mathrm{nec}}(1)\rvert, \lvert\mathrm{ATE}_{\mathrm{suf}}(0)\rvert)$	
	A	B	A	B
Flip rate	10.0	41.6	38.0	62.5

Problem: A square has a side length of 10 inches. Congruent isosceles right triangles are cut off each corner so that the resulting octagon has equal side lengths. How many inches are in the length of one side of the octagon? Express your answer as a decimal to the nearest hundredth.
CoT: ...Then, the side length of the octagon is $x\sqrt{2} = \frac{10}{\sqrt{2}+2} \cdot \sqrt{2} = \frac{10\sqrt{2}}{\sqrt{2}+2}$, which simplifies to $10\sqrt{2} - 10 \approx 4.142$ inches. So, 4.14 inches.
Wait, but let me compute $10\sqrt{2} - 10$ more accurately. $\sqrt{2} \approx 1.41421356$, so $10\sqrt{2} \approx 14.1421356$ and $14.1421356 - 10 = 4.1421356$ inches, which rounds to 4.14 inches.
So, the length of one side of the octagon is approximately 4.14.

Figure 13: An example of unfaithful self-verification steps (highlighted in blue) where the TTS score of each step is found smaller than 0.005. Low TTS indicates that those steps are not truly engaged in computation; rather, these reasoning steps are likely to be decorative and function as an appearance of self-verification, contributing minimally to the model's final prediction.

H Examples with low TTS

We provide more examples where reasoning steps have TTS lower than 0.005. For this section, we classify each reasoning example based on behavioral patterns labeled in Venhoff et al. (2025)

Uncertainty-estimation example

Context steps (last two sentences): Another way is to divide the quadrilateral into two triangles and find the area of each triangle, then add them up. Looking at the points, maybe I can split the quadrilateral along the x-axis

Steps with low TTS — Uncertainty-estimation: The points (-1, 0) and (2, 0) are on the x-axis, so the quadrilateral is split into two triangles: one above the x-axis and one below.
Wait, actually, the point (0, 1) is above the x-axis, and (0, -3) is below. So, if I connect (-1, 0) to (0, 1) to (2, 0) to (0, -3) and back to (-1, 0), it's a quadrilateral that crosses over the x-axis.
Alternatively, maybe I can split it into two triangles: one with vertices (-1, 0), (0, 1), (2, 0) and another with vertices (2, 0), (0, -3), (-1, 0). Wait, but that might not cover the entire quadrilateral.
Alternatively, perhaps split it into triangles with a common diagonal. Let me see.
If I connect (0, 1) to (0, -3), that's a vertical line along the y-axis. Then, the quadrilateral is split into two parts: one on the left side of the y-axis and one on the right.
But actually, the points are (-1, 0), (0, 1), (2, 0), (0, -3)

Uncertainty-estimation example

Context steps (last two sentences): Looking at the points, maybe I can split the quadrilateral along the x-axis. The points (-1, 0) and (2, 0) are on the x-axis, so the quadrilateral is split into two triangles: one above the x-axis and one below

Steps with low TTS — Uncertainty-estimation:
Wait, actually, the point (0, 1) is above the x-axis, and (0, -3) is below. So, if I connect (-1, 0) to (0, 1) to (2, 0) to (0, -3) and back to (-1, 0), it's a quadrilateral that crosses over the x-axis.
Alternatively, maybe I can split it into two triangles: one with vertices (-1, 0), (0, 1), (2, 0) and another with vertices (2, 0), (0, -3), (-1, 0). Wait, but that might not cover the entire quadrilateral.
Alternatively, perhaps split it into triangles with a common diagonal. Let me see.
If I connect (0, 1) to (0, -3), that's a vertical line along the y-axis. Then, the quadrilateral is split into two parts: one on the left side of the y-axis and one on the right.
But actually, the points are (-1, 0), (0, 1), (2, 0), (0, -3)

Backtracking example

Context steps (last two sentences): So, putting it all together, 20% of 50% of 80 is 8. That seems correct, but let me verify it another way to make sure I didn't make a mistake

Steps with low TTS — Backtracking:
Another approach is to multiply all the percentages together first and then apply them to 80. So, 20% is 0. 2, and 50% is 0. 5. Multiplying those together: 0. 2 * 0. 5 = 0. 1

Adding-knowledge example

Context steps (last two sentences): Wait, but that might not be accurate. Alternatively, maybe split the quadrilateral into two triangles by drawing a diagonal from (-1, 0) to (2, 0)

Steps with low TTS — Adding-knowledge: Then, the quadrilateral is split into two triangles: one with vertices (-1, 0), (0, 1), (2, 0) and another with vertices (-1, 0), (2, 0), (0, -3).
Let me calculate the area of each triangle and add them.
First triangle: (-1, 0), (0, 1), (2, 0)
Using the formula for the area of a triangle with coordinates:
1/2 |x1(y2 - y3) + x2(y3 - y1) + x3(y1 - y2)|
Plugging in the points:
x1 = -1, y1 = 0
x2 = 0, y2 = 1
x3 = 2, y3 = 0
So,
Area = 1/2 | (-1)(1 - 0) + 0(0 - 0) + 2(0 - 1) |
= 1/2 | (-1)(1) + 0 + 2(-1) |
= 1/2 | -1 + 0 - 2 |
= 1/2 | -3 | = 1/2 * 3 = 1

Adding-knowledge example

Context steps (last two sentences): Let me calculate the area of each triangle and add them. First triangle: (-1, 0), (0, 1), (2, 0)
Using the formula for the area of a triangle with coordinates:
1/2 |x1(y2 - y3) + x2(y3 - y1) + x3(y1 - y2)|
Plugging in the points:
x1 = -1, y1 = 0
x2 = 0, y2 = 1
x3 = 2, y3 = 0
So,
Area = 1/2 | (-1)(1 - 0) + 0(0 - 0) + 2(0 - 1) |
= 1/2 | (-1)(1) + 0 + 2(-1) |
= 1/2 | -1 + 0 - 2 |
= 1/2 | -3 | = 1/2 * 3 = 1.5

Steps with low TTS — Adding-knowledge: Second triangle: (-1, 0), (2, 0), (0, -3)
Again, using the same formula:
x1 = -1, y1 = 0
x2 = 2, y2 = 0
x3 = 0, y3 = -3
Area = 1/2 | (-1)(0 - (-3)) + 2((-3) - 0) + 0(0 - 0) |
= 1/2 | (-1)(3) + 2(-3) + 0 |
= 1/2 | -3 -6 + 0 |
= 1/2 | -9 | = 1/2 * 9 = 4. 5
Adding both areas: 1. 5 + 4. 5 = 6
Okay, so that's the same result as before