

Reasoning Beyond Chain-of-Thought: A Latent Computational Mode in Large Language Models

Zhenghao He Guangzhi Xiong Bohan Liu Sanchit Sinha Aidong Zhang
University of Virginia, USA

Abstract

Chain-of-Thought (CoT) prompting has improved the reasoning performance of large language models (LLMs), but it remains unclear why it works and whether it is the unique mechanism for triggering reasoning in large language models. In this work, we study this question by directly analyzing and intervening on the internal representations of LLMs with Sparse Autoencoders (SAEs), identifying a small set of latent features that are causally associated with LLM reasoning behavior. Across multiple model families and reasoning benchmarks, we find that steering a single reasoning-related latent feature can substantially improve accuracy without explicit CoT prompting. For large models, latent steering achieves performance comparable to standard CoT prompting while producing more efficient outputs. We further observe that this reasoning-oriented internal state is triggered early in generation and can override prompt-level instructions that discourage explicit reasoning. Overall, our results suggest that multi-step reasoning in LLMs is supported by latent internal activations that can be externally activated, while CoT prompting is one effective, but not unique, way of activating this mechanism rather than its necessary cause.

1 Introduction

Large language models exhibit substantially improved performance on complex reasoning tasks with Chain-of-Thought (CoT) prompting, where intermediate reasoning steps are explicitly verbalized (Wei et al., 2022; Kojima et al., 2022). Since its introduction, CoT prompting has consistently enhanced performance across a wide range of arithmetic (Lewkowycz et al., 2022), symbolic, and logical reasoning (Talmor et al., 2019) benchmarks and has inspired numerous variants, such as self-consistency (Wang et al., 2022). These successes have led to the widespread view that step-by-step prompt plays a central role in enabling multi-step reasoning in large language models.

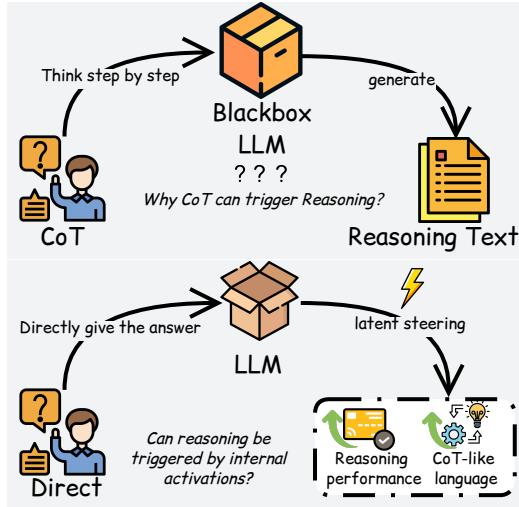


Figure 1: Multiple triggers for latent reasoning in LLMs. **Top:** CoT prompting produces explicit reasoning text, while the internal mechanism responsible for its effectiveness remains unclear. **Bottom:** We view reasoning as a latent internal mechanism that can be activated through different triggers, including latent steering.

However, the causal role of CoT prompting in multi-step reasoning remains unclear: CoT may be a convenient trigger rather than the unique pathway to reasoning behavior (Figure 1, top). Recent work shows that reasoning-relevant trajectories can be induced without CoT-style prompts, for example, by modifying the decoding process to surface latent CoT paths (Wang and Zhou, 2024), or by injecting continuous “soft thought” representations instead of generating explicit reasoning text (Xu et al., 2025). Moreover, causal analyses suggest that CoT traces are not necessarily the mechanism producing the final answer (Bao et al., 2024). These studies raise a fundamental question: *whether multi-step reasoning in LLMs corresponds to a latent internal mechanism that can be selectively activated, and whether CoT prompting is uniquely responsible for activating this mechanism or merely one of several effective triggers?* (Figure 1, bottom).

In this work, we answer this question by directly analyzing and intervening on the internal representations of large language models. Using Sparse Autoencoders (SAE) to identify latent features associated with reasoning behavior, we show that targeted modulation of these features can influence the model’s reasoning performance without explicit CoT prompting. Together, these findings suggest that multi-step reasoning reflects a latent capability inherent to the model, while CoT prompting is not the fundamental cause of this capability, but one of several ways to activate an underlying reasoning mechanism. Our contributions are:

- **Methodological:** We develop a two-stage pipeline using Sparse Autoencoders (SAEs) to identify reasoning-related latent features and causally validate their role through targeted steering interventions.
- **Empirical:** Experiments across six model families (up to 70B) demonstrate that steering a single latent feature at the first generation step matches or exceeds CoT performance while substantially reducing token overhead.
- **Mechanistic:** We show that this internal reasoning mode can be triggered early in generation and is robust enough to override prompt-level constraints like the \no_think instruction used in Qwen models.

2 Related Work

Chain-of-Thought and Multi-step Reasoning. Chain-of-thought (CoT) prompting has been widely adopted as a practical approach for improving performance on tasks that require multi-step reasoning (Wei et al., 2022). Prior work has demonstrated that encouraging models to produce intermediate reasoning steps can lead to substantial gains across a variety of domains, including arithmetic problem solving (Lewkowycz et al., 2022), symbolic manipulation, and logical inference (Talmor et al., 2019). As a result, CoT-style prompting has become a common component in reasoning benchmarks and evaluation protocols for large language models, and has inspired numerous extensions such as self-consistency (Wang et al., 2022) and structured reasoning prompts (Kojima et al., 2022).

Reasoning Beyond Explicit CoT Prompting. Beyond the standard CoT prompting paradigm, a growing line of work suggests that multi-step reasoning behavior in large language models need not be uniquely tied to explicit CoT prompts. For

instance, recent studies show that CoT-style reasoning trajectories can be elicited by altering the decoding process without using explicit prompting (Wang and Zhou, 2024), and that implicit reasoning leveraging internal hidden states can support complex reasoning without generating step-by-step text (Deng et al., 2023). Moreover, methods based on continuous or latent representations, such as soft thought tokens, demonstrate enhanced reasoning capability without relying on explicit verbal reasoning steps (Xu et al., 2025). Complementary empirical analyses further indicate that the effectiveness of CoT prompting does not strictly depend on correct or valid intermediate chains, suggesting that the internal drives for reasoning extend beyond the surface verbal structure (Wang et al., 2023).

Internal Representation Analysis of Reasoning. Beyond output-based analyses, prior work has investigated reasoning by examining internal representations of language models. Early approaches rely on probing, activation analysis, and causal tracing to associate hidden states with reasoning-relevant behaviors, suggesting that substantial computation occurs internally even when it is not explicitly verbalized (Burns et al.; Meng et al., 2022).

More recent work in mechanistic interpretability aims to decompose superposed activations into more interpretable components. Sparse autoencoders (SAEs) have been proposed as a scalable tool for extracting monosemantic or behaviorally meaningful features from language model activations, enabling finer-grained analysis of internal mechanisms (Cunningham et al., 2023). These representations have been used to study internal structure and to support activation-level interventions and steering, including in reasoning-related settings (Xin et al., 2025; Wang et al., 2025).

However, most existing studies remain primarily correlational, identifying internal features associated with reasoning-like behavior without establishing whether such representations play a causal role or correspond to a distinct reasoning mode that can be selectively engaged.

3 Method

In this section, we present a two-stage pipeline for identifying and intervening on latent features associated with reasoning-related behaviors in large language models. Chain-of-Thought prompting is used as a contrasting condition to reveal prompt-dependent differences in latent activations. As illus-

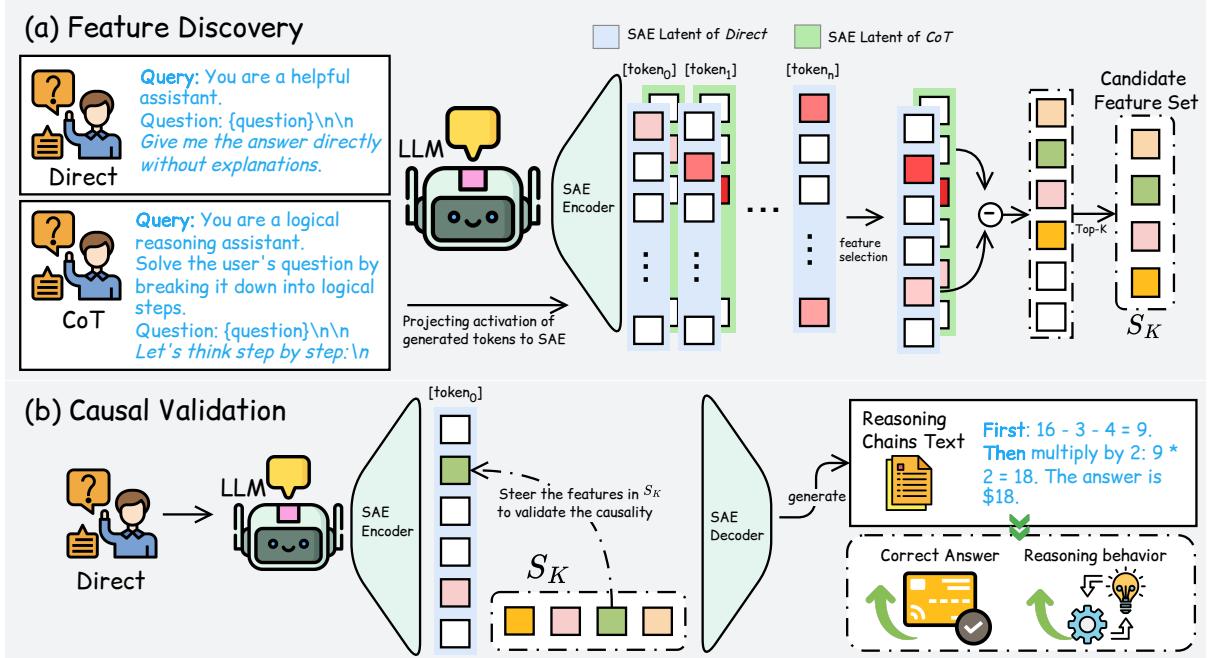


Figure 2: **Overview of the proposed two-stage pipeline.** (a) *Feature Discovery*. We contrast direct and chain-of-thought prompting to extract token-level activations, project them into sparse latent features using a pretrained sparse autoencoder (SAE), and identify prompt-sensitive candidate features via differential analysis. (b) *Causal Validation*. We apply targeted latent steering to selected features and inject the resulting residual into the model to assess their intervention sensitivity, evaluating the effect on model behavior and answer correctness.

trated in Figure 2, we first extract sparse latent features using a pretrained sparse autoencoder (SAE) and identify candidate features (Section 3.3). We then define a latent steering procedure to estimate the intervention sensitivity of individual features on training data, and evaluate the selected features on a held-out test set (Section 3.4).

3.1 Problem Setup

We consider a pretrained large language model f_θ with fixed parameters. Given an input question x and a prompting condition p , the model generates an output sequence autoregressively. We focus on two prompting regimes: direct prompting p^{dir} and chain-of-thought prompting p^{CoT} , which differ only in their instructions:

$$y \sim f_\theta(x, p), \quad p \in \{p^{\text{dir}}, p^{\text{CoT}}\}. \quad (1)$$

Let $h_{l,t}^{(p)} \in \mathbb{R}^d$ denote the hidden activation at layer l and token position t under prompting condition p , where

$$h_{l,t}^{(p)} = f_\theta^{(l)}(x, p, y_{<t}). \quad (2)$$

Our analysis focuses on latent activations aggregated from the generation process, with the specific aggregation strategy selected empirically and described in later sections.

3.2 Latent Feature Extraction with SAE

To obtain a sparse and intervention-friendly latent representation of model activations, we employ a pretrained SAE. For a given hidden activation $h_{l,t}^{(p)} \in \mathbb{R}^d$ at layer l and token position t , the SAE encoder maps it to a latent vector $z_{l,t}^{(p)} \in \mathbb{R}^m$:

$$z_{l,t}^{(p)} = \phi(W_{\text{enc}} h_{l,t}^{(p)} + b_{\text{enc}}), \quad (3)$$

where $\phi(\cdot)$ denotes an element-wise activation function, depending on the specific SAE variant. The corresponding decoder reconstructs the activation as

$$\hat{h}_{l,t}^{(p)} = W_{\text{dec}} z_{l,t}^{(p)} + b_{\text{dec}}. \quad (4)$$

The SAE is trained to minimize a reconstruction objective of the form

$$\mathcal{L}_{\text{SAE}} = \mathbb{E} \left[\left\| h_{l,t}^{(p)} - \hat{h}_{l,t}^{(p)} \right\|_2^2 \right] + \lambda \Omega(z_{l,t}^{(p)}), \quad (5)$$

where $\Omega(\cdot)$ denotes a sparsity-inducing regularizer on the latent activations, such as an ℓ_1 penalty or a top- k masking constraint, depending on the SAE variant. In this work, we use a fixed, pretrained SAE and keep its parameters frozen during the analysis.

3.3 Differential Feature Discovery

We identify candidate latent features by contrasting their activation statistics under different prompting regimes. For a fixed input distribution over questions $x \sim \mathcal{D}$, direct prompting and chain-of-thought prompting induce two conditional distributions over latent activations.

Feature Aggregation. Given a single generated sequence under the prompting condition p , let $\{\tilde{z}_{l,t}^{(p)}(x)\}_{t=1}^T$ denote the SAE latent activations extracted at layer l across token positions. We first apply an aggregation function $g(\cdot)$ to obtain a fixed-dimensional representation for each example:

$$\tilde{z}^{(p)}(x) = g\left(\{\tilde{z}_{l,t}^{(p)}(x)\}_{t=1}^T\right), \quad (6)$$

where $g(\cdot)$ denotes a fixed aggregation over token-level activations. In our main experiments, we select the latent representation at the first generation step. We also evaluated alternative aggregation strategies, including average pooling and max pooling over the generated sequence, but found them to be consistently less effective.

We hypothesize that latent features associated with CoT-style behavior are primarily activated early in generation, before the model converges toward the final answer. Aggregating over later tokens may therefore dilute these signals, consistent with prior findings that reasoning-related activations emerge early in the generation process (David, 2025).

Candidate Feature Set. We then define the conditional mean activation of latent feature k under prompting condition p as the empirical average of its aggregated activation across a dataset of inputs:

$$\mu_k(p) = \mathbb{E}_{x \sim \mathcal{D}} \left[\tilde{z}_k^{(p)}(x) \right] = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \tilde{z}_k^{(p)}(x). \quad (7)$$

Using these statistics, we compute a feature-wise differential score that captures prompt-induced modulation:

$$\Delta z_k = \mu_k(p^{\text{CoT}}) - \mu_k(p^{\text{dir}}). \quad (8)$$

We select latent features that are most sensitive to changes in the prompting strategy by choosing the K dimensions with the largest absolute differential scores. Formally, we define the selected feature set as

$$S_K = \arg \max_{S \subseteq \{1, \dots, m\}, |S|=K} \sum_{k \in S} |\Delta z_k|. \quad (9)$$

This procedure identifies a compact set of prompt-sensitive latent features, without introducing additional modeling assumptions or learned probes.

3.4 Causal Validation via Latent Steering

Given the candidate features S_K identified in Section 3.3, we define a latent steering procedure for a controlled intervention on model activations.

Let $a_{l,t}(x) \in \mathbb{R}^m$ denote the pre-activation latent representation produced by the SAE encoder at layer l and generation step t . The corresponding post-activation latent is given by $z_{l,t}(x) = \phi(a_{l,t}(x))$. For a chosen intervention set $S \subseteq S_K$, we define an intervention by modifying the corresponding pre-activation dimensions:

$$\tilde{a}_{l,t,k}^{\text{steer}}(x; S) = \begin{cases} a_{l,t,k}(x) + \alpha \mathbb{E}[|a_{l,t,k}(x)|], & k \in S, \\ a_{l,t,k}(x), & k \notin S, \end{cases} \quad (10)$$

where α is a scalar steering coefficient that controls the intervention strength.

We adopt an additive intervention on pre-activation latents to externally activate features, rather than amplify their current values, allowing the intervention to act even when a feature is inactive. The perturbation is normalized by the feature's empirical activation scale, ensuring comparable intervention strength across models.

The steered latent representation is obtained as

$$\tilde{z}_{l,t}^{\text{steer}}(x) = \phi(\tilde{a}_{l,t}^{\text{steer}}(x; S)). \quad (11)$$

The modified latent representation $\tilde{z}_{l,t}^{\text{steer}}(x)$ is mapped back to the activation space using the SAE decoder,

$$\hat{h}_{l,t}^{\text{steer}}(x) = W_{\text{dec}} \tilde{z}_{l,t}^{\text{steer}}(x) + b_{\text{dec}}. \quad (12)$$

Because the sparse autoencoder produces imperfect reconstruction, directly injecting the decoded steered representation may introduce reconstruction bias. Therefore, we apply a residual injection scheme. Let $\hat{h}_{l,t}(x) = \text{Dec}(z_{l,t}(x))$ denote the reconstruction of the original activation. The activation injected into the model is given by

$$\tilde{h}_{l,t}(x) = h_{l,t}(x) + (\hat{h}_{l,t}^{\text{steer}}(x) - \hat{h}_{l,t}(x)). \quad (13)$$

This residual correction ensures a localized intervention that isolates the effect of latent steering with minimal side effects.

Rather than intervening on all candidate features simultaneously, we estimate the intervention sensitivity of individual features on a held-out training set by restricting to singleton interventions

Model	Strategy	GSM8K		GPQA		BBH	
		(Cobbe et al., 2021)	Acc. (\uparrow)	#Tok (\downarrow)	(Rein et al., 2024)	Acc. (\uparrow)	#Tok (\downarrow)
LLaMA-3.1-8B-Instruct (Meta, 2024)	Direct	24.5	17 \pm 34	28.7	2 \pm 0	50.8	3 \pm 0
	Steered Direct	73.3	83 \pm 53	28.2	43 \pm 103	61.6	53 \pm 29
	CoT	79.3	234 \pm 70	20.7	423 \pm 98	77.2	146 \pm 28
	Steered CoT	84.1	227 \pm 68	22.2	414 \pm 99	78.4	146 \pm 27
LLaMA-3.3-70B-Instruct (Meta, 2025)	Direct	46.7	12 \pm 22	41.9	2 \pm 0	94.0	7 \pm 7
	Steered Direct	88.8	53 \pm 30	44.9	47 \pm 120	93.6	40 \pm 31
	CoT	96.1	268 \pm 63	19.7	495 \pm 41	100.0	226 \pm 44
	Steered CoT	96.5	257 \pm 66	18.6	474 \pm 66	100.0	228 \pm 39
Qwen3-0.6B (Team, 2025)	Direct	7.9	14 \pm 15	26.7	7 \pm 4	39.6	13 \pm 3
	Steered Direct	60.6	357 \pm 124	24.7	237 \pm 224	66.4	301 \pm 309
	CoT	59.7	400 \pm 108	15.6	508 \pm 18	82.8	537 \pm 431
	Steered CoT	59.6	393 \pm 109	16.7	506 \pm 24	78.4	559 \pm 487
Qwen3-4B (Team, 2025)	Direct	34.5	32 \pm 25	32.8	8 \pm 5	83.2	13 \pm 2
	Steered Direct	60.0	141 \pm 127	27.7	418 \pm 171	88.4	74 \pm 138
	CoT	61.1	421 \pm 101	18.8	510 \pm 12	98.8	458 \pm 328
	Steered CoT	62.5	422 \pm 101	19.2	511 \pm 10	97.6	528 \pm 348
Gemma-3-4B-Instruct (Team et al., 2025)	Direct	8.4	6 \pm 17	30.8	3 \pm 0	59.6	5 \pm 0
	Steered Direct	74.0	243 \pm 158	21.7	353 \pm 188	65.6	191 \pm 203
	CoT	78.1	193 \pm 92	26.2	291 \pm 219	50.0	161 \pm 155
	Steered CoT	82.8	232 \pm 104	19.2	415 \pm 147	80.8	187 \pm 122
Gemma-3-12B-Instruct (Team et al., 2025)	Direct	18.2	5 \pm 2	32.3	2 \pm 0	86.4	4 \pm 0
	Steered Direct	70.9	101 \pm 98	33.3	18 \pm 26	95.6	36 \pm 18
	CoT	92.7	181 \pm 75	24.4	404 \pm 133	95.2	136 \pm 70
	Steered CoT	92.8	168 \pm 71	22.2	400 \pm 137	96.8	120 \pm 58

Table 1: Reasoning performance across models and benchmarks under different strategies. For each model, we compare direct prompt, Chain-of-Thought (CoT) prompting, and their steered variants. Latent steering is applied only at the first generation step along an identified reasoning-related feature, with all subsequent decoding steps left unchanged. Accuracy (%) and average number of generated tokens (#Tok) are reported for each setting.

$S = \{k\}$ for each $k \in S_K$. For each candidate feature, we apply the latent steering procedure and measure the resulting change. Features with consistently positive intervention sensitivity are selected. The features are selected based on the training data and are fixed during evaluation.

4 Experiments

4.1 Implementation Details

For each model, we identify reasoning-related SAE features at a single mid-to-late transformer layer and perform steering at the same layer throughout all experiments. Since models differ in depth and architecture, the selected layer is model-specific and chosen based on where reasoning-related features most consistently emerge in practice. The steering strength α is selected according to the validation results in Section 3.4, with values in the range of 15–25 yielding stable behavior across models. Once determined, the steering layer, feature index, and α are fixed for each model and used

consistently across all datasets. Model-specific configurations are provided in Appendix A.6.

Datasets. We evaluate our method on four reasoning benchmarks: GSM8K (Cobbe et al., 2021), GPQA (Rein et al., 2024), and Big-Bench Hard (BBH) (Suzgun et al., 2022). For each base model, reasoning-related features are identified using 1,000 question–answer pairs randomly sampled from the *training split of GSM8K*. All reported evaluation results are obtained by steering the identified features on the GSM8K test set, the GPQA Diamond subset, and the logical_deduction_three_objects task from Big-Bench Hard (BBH).

Models. We study six language models of varying sizes: LLaMA-3-8B-Instruct, LLaMA-3-70B-Instruct, Qwen-3-0.6B, Qwen-3-4B, Gemma-3-4B-Instruct, and Gemma-3-12B-Instruct. All models are used in inference-only mode, with weights frozen throughout all experiments.

Sparse Autoencoders. We employ pretrained SAEs from Goodfire (Goodfire, 2024) for LLaMA

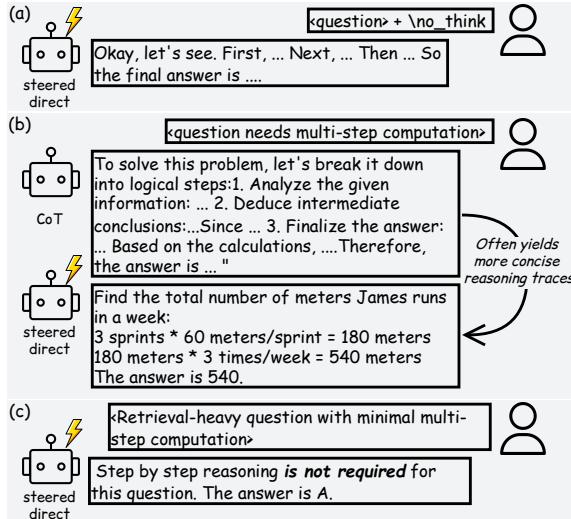


Figure 3: Behaviors induced by steering. (a) Steering can override prompt-level instructions that suppress reasoning (e.g., \no_think). (b) For questions requiring multi-step computation, steering can yield shorter explicit reasoning traces compared to standard chain-of-thought prompting. (c) On retrieval-heavy questions, latent steering leads the model to explicitly note that step-by-step reasoning is unnecessary. *These examples are illustrative and not intended to be exhaustive.*

models and GemmaScope (Lieberum et al., 2024) for Gemma models. For Qwen models, we train SAEs from scratch; training details are provided in Appendix A.4.

4.2 Effects of Latent Steering on Reasoning Performance

For each base model, we first identify a small set of reasoning-related latent features using the procedure described in Section 3.3 and Section 3.4. The specific feature indices selected for each model and the corresponding α are reported in Appendix A.5.

During evaluation, we apply latent steering only at the *first generation step* along the identified reasoning feature, following the step-wise intervention strategy introduced in Section 3.4. All other model components, decoding parameters, and prompts are kept identical to the unsteered baselines. The resulting performance is shown in Table 1.

Steered direct prompting improves reasoning accuracy. Compared to direct prompting, steered direct yields large accuracy gains on reasoning-intensive benchmarks such as GSM8K and BBH. This indicates that the identified latent features are closely tied to internal reasoning processes. Notably, steering can induce reasoning behavior even when the prompt explicitly discourages interme-

diate reasoning. For example, on Qwen models, steered direct decoding produces coherent multi-step reasoning despite prompts containing the control token \no_think, which is designed to suppress the explicit generation of CoT (Figure 3a). This behavior suggests that the intervention operates at the level of internal computation rather than surface prompt compliance.

Early single-feature steering leads to more concise reasoning outputs in large models. For larger models such as LLaMA-3.3-70B, steering achieves accuracy comparable to or exceeding standard CoT prompting while substantially reducing the length of generated reasoning text (Figure 3b). One possible explanation is that the SAE partially disentangles latent features associated with internal reasoning computation from those involved in verbalization. In contrast, standard CoT prompting may more strongly couple internal reasoning computation with explicit step-by-step verbalization, leading to longer reasoning traces. We emphasize that for this model, steering is applied to a single latent feature (feature #13709), highlighting the effectiveness of minimal, targeted interventions.

Steering provides limited benefit once the model is already in a reasoning mode. In many cases, steered CoT yields only marginal improvements or no improvement over standard CoT. We hypothesize that when the reasoning-related feature is already strongly activated under CoT prompting, additional amplification does not further enhance reasoning performance. This observation is consistent with the view that steering primarily acts as a trigger for entering a reasoning mode, rather than refining an ongoing reasoning process, as further analyzed in Section 4.4.

Limited gains on tasks with weak reliance on multi-step reasoning. On benchmarks such as GPQA, where CoT prompting does not consistently outperform direct decoding, steering likewise provides little or no improvement. In some cases, models refuse to do reasoning and directly produce answers when we strongly activate those reasoning features (Figure 3c). This suggests that when tasks do not strongly require multi-step reasoning, the identified features are not substantially engaged, and steering has limited effect.

4.3 Feature Dynamics During Generation

Figure 4 visualizes the activation dynamics of the reasoning-related SAE features used for steering during answer generation on GSM8K; these same

single latent dimensions are intervened on to obtain all steered results reported in Table 1. Across both models, the identified features exhibit a highly non-uniform temporal profile, with activations peaking early in decoding and rapidly decaying thereafter. This transient pattern suggests that the feature is primarily engaged during the initial phase of reasoning rather than throughout generation.

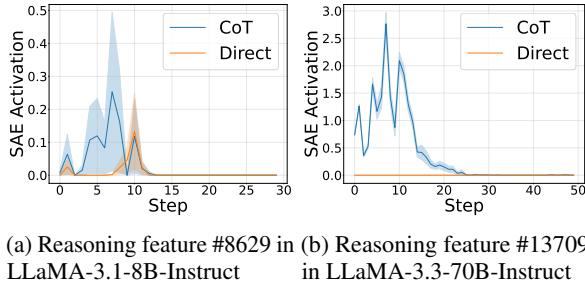


Figure 4: Activation dynamics of a reasoning-related SAE feature during generation on GSM8K.

Moreover, CoT prompting consistently induces stronger activation of the reasoning feature compared to direct decoding, indicating that the feature is associated with reasoning-oriented generation modes. Despite differences in scale, this temporal structure is qualitatively consistent across model sizes, with larger models exhibiting more concentrated and higher-magnitude activation. These observations support our focus on early interventions when steering reasoning-related latent features.

4.4 Reasoning Feature as a Mode Indicator

We analyze whether the identified feature is associated with *entering a reasoning mode* or with *reasoning quality*. Specifically, we examine the correlation between feature activation and (i) prompting strategy (CoT vs. direct), and (ii) answer correctness within each strategy. Figure 4 shows a characteristic early activation pattern during answer generation. We quantify this pattern by comparing the maximum activation within the first 20 decoding steps under CoT and direct prompting.

As shown in Table 2, we find that the feature is activated significantly more strongly under CoT prompting than under direct prompting. Across 200 GSM8K samples, the activation of the feature is substantially higher for CoT than for direct prompting (point-biserial $r = 0.14$, $p = 0.006$), indicating a strong association with entering a reasoning-oriented generation mode. Computation details are provided in Appendix A.7.

Target Variable	corr. (p -value)	Mean (A)	Mean (B)
Reasoning Mode (A=CoT, B=Direct)	0.14 (0.006)	0.10 ± 0.41	0.01 ± 0.13
CoT Acc. (A=Correct, B=Wrong)	-0.02 (0.80)	0.10 ± 0.40	0.12 ± 0.43
Direct Acc. (A=Correct, B=Wrong)	-0.06 (0.40)	0.00 ± 0.00	0.02 ± 0.13

Table 2: Analysis of the correlation between feature activation values and different target variables. Activation values are collected from feature #8629 in LLaMA-3.1-8B-Instruct on GSM8K. We report the correlation coefficients (with p -values) and the mean feature scores corresponding to different groups in the target variable.

In contrast, when conditioning on a fixed prompting strategy (either CoT or direct), feature activation shows little correlation with answer correctness. As shown in Table 2, the feature does not reliably distinguish correct from incorrect answers under either strategy.

This interpretation also explains the limited gains observed when steering the feature under CoT prompting in Table 1: *once the model has already entered a reasoning mode, further amplifying the trigger alone does not substantially improve reasoning performance.*

4.5 Effect of Intervention Timing

We analyze how the effectiveness of steering depends on the timing of the intervention during generation. Following the feature ranking defined in Section 3.3, we consider both an intervention on the top-ranked feature (feature #8629 in LLaMA-3.1-8B-Instruct) and, for comparison, an intervention on the top-10 reasoning-related features. In all cases, we apply the intervention at a single decoding step while varying the step index.

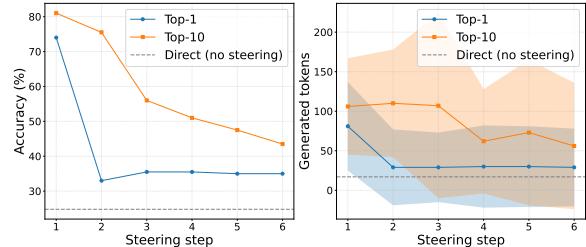


Figure 5: Effect of intervention timing when steering reasoning-related latent features on GSM8K with LLaMA-3.1-8B-Instruct. The figure reports accuracy (left) and generated token length (right) as a function of the decoding step at which the intervention is applied.

As shown in Figure 5, earlier interventions consistently yield stronger effects. Intuitively, entering a reasoning mode early gives the model more opportunity to carry out the intermediate computations needed for a correct solution, whereas late

interventions may occur when the model is already close to, or has effectively committed to, an answer. This timing consideration is important for direct prompting, where some responses terminate within only a few tokens (e.g., ~ 3 tokens), making later intervention steps ill-defined or ineffective.

Comparing top-1 and top-10 steering, both show a similar dependence on intervention timing, with earlier interventions being more effective, but their behaviors differ. As the intervention is delayed, top-10 steering weakens gradually, whereas top-1 steering drops sharply after the first step and quickly approaches the unsteered baseline. This suggests that a single feature can trigger reasoning when applied early, while sustained reasoning and its verbalization depend on multiple latent features, which is also reflected in the longer generations produced by top-10 steering.

4.6 Cross-Prompt Activation of Reasoning Features

We next examine whether the identified reasoning-related feature is tied to a specific CoT prompt, or whether it generalizes across different ways of encouraging reasoning. Figure 6 shows the activation of the identified reasoning-related feature under different prompting conditions. Beyond explicit step-by-step prompting, we observe that several alternative prompts that implicitly encourage reasoning (e.g., “Explain how”, “Solve carefully”, and “Think”) consistently activate the same feature, in some cases to an even greater extent (detailed prompts can be found in Appendix A.3). This indicates that the feature is not tied to a specific prompt template, but instead reflects a more general internal reasoning mode that can be triggered by diverse linguistic cues.

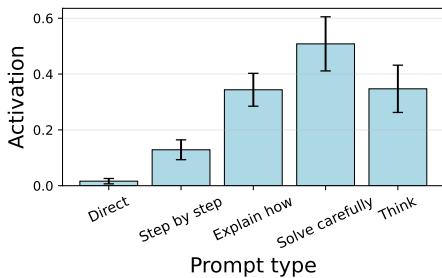


Figure 6: Early activation of the reasoning-related SAE feature under different prompting conditions on GSM8K using LLaMA-3.1-8B-Instruct.

At the same time, as discussed in Section 4.4, activation of this feature alone is not sufficient

to predict answer correctness. Despite exhibiting comparable or stronger activation than step-by-step prompting, these alternative prompts achieve similar accuracy on GSM8K. Together, these results suggest that the identified feature primarily functions as a trigger for entering a reasoning mode, while successful problem solving depends on additional factors beyond the activation strength of this single latent dimension.

4.7 Ablation Study

We conduct a random feature ablation to verify that the observed steering effects are specific to the identified reasoning-related latent feature rather than arising from arbitrary perturbations. Using LLaMA-3.1-8B-Instruct under direct prompting, we compare steering the reasoning feature (#8629) with steering randomly selected non-zero SAE features from the same layer.

Feature	Acc. (%)	#Tok	Runs
#8629	73.3	83	-
Random	26.1 \pm 3.4	29 \pm 5	10

Table 3: Random feature ablation on GSM8K with LLaMA-3.1-8B-Instruct. Results for the reasoning-related feature are reported with a fixed random seed.

As shown in Table 3, steering the reasoning-related feature yields substantial gains in accuracy and induces longer generations, whereas random feature steering fails to produce comparable improvements. This result confirms that the steering effect is specific to the identified reasoning-related feature rather than a generic latent intervention.

5 Conclusion

In this work, we show that multi-step reasoning in large language models can be triggered by activating a latent internal mechanism. By analyzing and intervening on latent representations with sparse autoencoders, we identify reasoning-related features whose activation induces reasoning behavior even under direct prompting. Experiments across multiple models and benchmarks demonstrate that direct latent steering improves reasoning performance while often producing shorter outputs than Chain-of-Thought prompting. These results suggest that Chain-of-Thought prompting is one effective, but not unique, way to trigger reasoning, and highlight the usefulness of activation-level interventions for understanding and controlling reasoning in language models.

Limitations

Although our results are consistent with the interpretation that SAE representations partially disentangle latent features related to internal reasoning computation from those associated with verbose verbalization, we do not claim that the identified features constitute fully disentangled or isolated reasoning mechanisms. Our conclusions are based on behavioral and intervention-level evidence rather than complete mechanistic characterization.

References

- Guangsheng Bao, Hongbo Zhang, Linyi Yang, Cunxiang Wang, and Yue Zhang. 2024. Llms with chain-of-thought are non-causal reasoners. *CoRR*.
- Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, and 1 others. 2025. Smollm2: When smol goes big – data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*.
- Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering latent knowledge in language models without supervision, 2024. URL <https://arxiv.org/abs/2212.03827>.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. 2023. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*.
- Joey David. 2025. Temporal predictors of outcome in reasoning language models. *arXiv preprint arXiv:2511.14773*.
- 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*.
- Goodfire. 2024. Sparse autoencoders for llama-3 models. <https://huggingface.co/Goodfire/Llama-3.3-70B-Instruct-SAE-150>. Accessed: 2025-01.
- 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.
- Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, and 1 others. 2022. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857.
- Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. 2024. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. *arXiv preprint arXiv:2408.05147*.
- Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*.
- Meta. 2024. **Llama 3.1: Open foundation and instruction-tuned large language models.** Llama-3.1-8B-Instruct model.
- Meta. 2025. **meta-llama/llama-3.3-70b-instruct.** Llama-3.3-70B-Instruct model.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. **GPQA: A graduate-level google-proof q&a benchmark.** In *First Conference on Language Modeling*.
- Mirac Suzgun, Nathan Scales, Nathanael Schärfli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.
- 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.
- Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, and 1 others. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.
- Qwen Team. 2025. **Qwen3 technical report.** Preprint, arXiv:2505.09388.
- Anyi Wang, Dong Shu, Yifan Wang, Yunpu Ma, and Mengnan Du. 2025. Improving llm reasoning through interpretable role-playing steering. *arXiv preprint arXiv:2506.07335*.
- Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2023. Towards understanding chain-of-thought prompting:

An empirical study of what matters. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 2717–2739.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Xuezhi Wang and Denny Zhou. 2024. Chain-of-thought reasoning without prompting. *Advances in Neural Information Processing Systems*, 37:66383–66409.

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.

Chunlei Xin, Shuheng Zhou, Huijia Zhu, Weiqiang Wang, Xuanang Chen, Xinyan Guan, Yaojie Lu, Hongyu Lin, Xianpei Han, and Le Sun. 2025. Sparse latents steer retrieval-augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4547–4562.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. 2025. Softcot: Soft chain-of-thought for efficient reasoning with llms. *arXiv preprint arXiv:2502.12134*.

A Appendix

A.1 Dataset Details

A.1.1 GSM8K

GSM8K is a benchmark of grade-school level mathematical word problems that require multi-step numerical reasoning (Cobbe et al., 2021). Each example consists of a natural language question and a short numerical answer. Solving these problems typically involves parsing the problem description, performing a sequence of arithmetic operations, and producing a final numeric result.

In our experiments, GSM8K is used both for identifying reasoning-related latent features and for evaluating reasoning performance. Specifically, we randomly sample 1,000 examples from the GSM8K training split to identify candidate reasoning-related features using the procedure described in Section 3.3. All reported evaluation results are obtained on the GSM8K test split. Accuracy is measured as exact match between the model output and the ground-truth answer, following standard evaluation protocols.

A.1.2 GPQA

GPQA is a challenging question answering benchmark designed to assess scientific reasoning at the graduate level (Rein et al., 2024). The dataset covers multiple domains, including physics, chemistry, and biology, and is constructed to be resistant to surface-level retrieval by large language models. Questions often require combining domain knowledge with logical reasoning rather than simple factual recall.

We evaluate our method on the GPQA Diamond subset, which contains questions with verified difficulty and high annotation quality. Model performance is measured by exact match accuracy over multiple-choice answers. We note that prior work has shown that chain-of-thought prompting does not consistently improve performance on GPQA, making it a useful testbed for examining whether latent steering selectively benefits tasks that rely on multi-step reasoning.

A.1.3 Big-Bench Hard

Big-Bench Hard (BBH) is a curated subset of the BIG-Bench benchmark that focuses on tasks known to be challenging for language models (Suzgun et al., 2022). The tasks in BBH are selected based on their low performance under standard prompting and often require logical deduction, symbolic

manipulation, or multi-step reasoning.

In this work, we evaluate on the `logical_deduction_three_objects` task from BBH, which requires reasoning about relational constraints among multiple entities. This task emphasizes structured logical reasoning rather than numerical computation. Model performance is evaluated using exact match accuracy on the final answer. BBH serves as a complementary benchmark to GSM8K, allowing us to test whether the identified reasoning-related features generalize beyond arithmetic reasoning to logical deduction tasks.

A.2 Model Details

We evaluate our method on six pretrained, instruction-tuned large language models drawn from three widely used model families: LLaMA-3, Qwen-3, and Gemma-3. These families differ in architecture, training data, and scale, allowing us to assess the generality of latent steering across diverse model designs. All models are used in inference-only mode, with parameters frozen throughout all experiments.

A.2.1 LLaMA-3 Family

We consider two instruction-tuned models from the LLaMA-3 family: LLaMA-3.1-8B-Instruct and LLaMA-3.3-70B-Instruct. These models represent small- and large-scale regimes within the same architectural family, enabling controlled comparisons across model size. LLaMA-3 models are widely adopted as strong open-weight baselines for reasoning tasks, and prior work has shown that chain-of-thought prompting is particularly effective for larger LLaMA models. We use pretrained sparse autoencoders released by Goodfire for both LLaMA-3 models.

A.2.2 Qwen-3 Family

We evaluate two models from the Qwen-3 family: Qwen-3-0.6B and Qwen-3-4B. These models provide additional coverage of smaller-scale language models and differ substantially from LLaMA-3 in training data composition and architectural choices. For Qwen-3 models, pretrained sparse autoencoders are not publicly available, so we train sparse autoencoders from scratch following the procedure described in Appendix A.4. Including Qwen-3 allows us to examine whether reasoning-related latent features and steering effects persist in models with more limited capacity.

A.2.3 Gemma-3 Family

We use two instruction-tuned models from the Gemma-3 family: Gemma-3-4B-Instruct and Gemma-3-12B-Instruct. Gemma-3 models are trained with different data and optimization strategies compared to LLaMA-3 and Qwen-3, offering an additional axis of architectural and training diversity. We employ pretrained sparse autoencoders from GemmaScope for both Gemma-3 models. Results on Gemma-3 help assess the robustness of latent steering across independently developed model families.

A.3 Input Prompts

We use fixed prompt templates for all experiments to ensure comparability across models and conditions. For each model family, we define a *direct* prompt and a *chain-of-thought (CoT)* prompt that differ only in whether explicit step-by-step reasoning is encouraged. Unless otherwise specified, all prompts are applied consistently across datasets, with decoding parameters held fixed. Due to the length of the prompt templates and the presence of model-specific special tokens, we present the full prompts as figures for readability.

A.3.1 LLaMA-3 Family

For models in the LLaMA-3 family, we use fixed direct and chain-of-thought prompt templates. The full prompt templates are shown in Figures 7 and 8.

Direct Prompt for LLaMA Family

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\n
You are a helpful assistant.<|eot_id|>
<|start_header_id|>user<|end_header_id|>\n\n
Question: {question}\n\n
Give me the answer directly without explanations.<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>\n\n
```

Figure 7: Direct prompt template for the LLaMA-3 family.

CoT Prompt for LLaMA Family

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\n
You are a logical reasoning assistant. Solve the user's question by
breaking it down into logical steps.
Finally, provide the answer in the specified format.<|eot_id|>
<|start_header_id|>user<|end_header_id|>\n\n
Question: {question}\n\n
Let's think step by step:\n
    1) Analyze the given information.\n
    2) Deduce intermediate conclusions.\n
    3) Finalize the answer.\n\n
Format your final response as: 'Therefore, the answer is [your
answer].'<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>\n\n"
```

Figure 8: Chain-of-thought prompt template for the LLaMA-3 family.

In addition to the direct and CoT prompts, we also consider several alternative prompts for the cross-prompt analysis in Section 4.6. These prompts implicitly encourage careful reasoning without explicitly requesting step-by-step explanations. The full templates for these prompts are shown in Figures 9–11.

Think Prompt. See Figure 9 for the full template.

Think Prompt for LLaMA Family

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\n
You are a helpful assistant. Think carefully before answering.
Finally, provide the answer in the specified format.<|eot_id|>
<|start_header_id|>user<|end_header_id|>\n\n
Question: {question}\n\n
Think carefully before answering.\n\n
Format your final response as: 'Therefore, the answer is [your
answer].'<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>\n\n"
```

Figure 9: Think prompt template for the LLaMA-3 family.

Explain Prompt. See Figure 10 for the full template.

Explain Prompt for LLaMA Family
<pre>< begin_of_text >< start_header_id >system< end_header_id >\n\n You are a helpful assistant. Explain how to solve the user's question. Finally, provide the answer in the specified format.< eot_id > < start_header_id >user< end_header_id >\n\n Question: {question}\n\n Explain how you get the answer.\n\n: Format your final response as: 'Therefore, the answer is [your answer].< eot_id > < start_header_id >assistant< end_header_id >\n\n"</pre>

Figure 10: Explain prompt template for the LLaMA-3 family.

Solve Carefully Prompt. See Figure 11 for the full template.

Solve Carefully Prompt for LLaMA Family
<pre>< begin_of_text >< start_header_id >system< end_header_id >\n\n You are a helpful assistant. Solve the user's question carefully. Finally, provide the answer in the specified format.< eot_id > < start_header_id >user< end_header_id >\n\n Question: {question}\n\n Solve the problem carefully.\n\n: Format your final response as: 'Therefore, the answer is [your answer].< eot_id > < start_header_id >assistant< end_header_id >\n\n"</pre>

Figure 11: Solve carefully prompt template for the LLaMA-3 family.

A.3.2 Qwen-3 Family

For the Qwen-3 family, we define direct and chain-of-thought prompts analogous to those used for LLaMA-3, adapted to the instruction format expected by Qwen models. The full prompt templates are shown in Figures 12 and 13.

Direct Prompt. See Figure 12 for the full template.

Direct Prompt for Qwen Family
<pre>< im_start >system You are a helpful assistant.< im_end > < im_start >user {question} Give me the answer directly without explanations. Give only the final answer. Do not include reasoning, analysis, or intermediate steps. Output a single sentence only./no_think < im_end > < im_start >assistant\n</pre>

Figure 12: Direct prompt template for the Qwen-3 family.

Chain-of-Thought Prompt. See Figure 13 for the full template.

CoT Prompt for Qwen Family
<pre>< im_start >system You are a logical reasoning assistant. Solve the user's question by breaking it down into logical steps. Finally, provide the answer in the specified format.< im_end > < im_start >user {question} Let's think step by step: 1) Analyze the given information. 2) Deduce intermediate conclusions 3) Finalize the answer. Format your final response as: 'Therefore, the answer is [your answer]./think< im_end > < im_start >assistant</pre>

Figure 13: Chain-of-thought prompt template for the Qwen-3 family.

A.3.3 Gemma-3 Family

For the Gemma-3 family, we likewise use a direct prompt and a chain-of-thought prompt that follow the instruction conventions of Gemma models. The full templates are shown in Figures 14 and 15.

Direct Prompt. See Figure 14 for the full template.

Direct Prompt for Gemma Family
<pre>{question}\n Give me the answer directly without explanations.<eos></pre>

Figure 14: Direct prompt template for the Gemma-3 family.

Chain-of-Thought Prompt. See Figure 15 for the full template.

CoT Prompt for Gemma Family
<pre>You are a logical reasoning assistant. Solve the user's question by breaking it down into logical steps. Finally, provide the answer in the specified format. Question: {question}\n Let's think step by step: 1) Analyze the given information. 2) Deduce intermediate conclusions. 3) Finalize the answer.\n Format your final response as: 'Therefore, the answer is [your answer]. <eos></pre>

Figure 15: Chain-of-thought prompt template for the Gemma-3 family.

Prompt Consistency. Across all model families, the direct and CoT prompts differ only in their instructions regarding explicit reasoning. No task-specific information or answer hints are introduced

Model	Steered Feature Index	Steering Step	Steering Strength α	Steering Layer
LLaMA-3.1-8B-Instruct (Meta, 2024)	#8629	1	15	19
LLaMA-3.3-70B-Instruct (Meta, 2025)	#13709	1	20	50
Qwen3-0.6B (Team, 2025)	#15352, #14424, #27058, #22718, #12509	1	25	19
Qwen3-4B (Team, 2025)	#12714, #66391, #58281, #60128, #25787	1	70	29
Gemma-3-4B-Instruct (Team et al., 2025)	#113617, #80327, #165330, #52856, #923, #51564, #8253, #6879, #1257, #5239	1	15	29
Gemma-3-12B-Instruct (Team et al., 2025)	#6662, #7847, #97, #15521, #5100, #32828, #2616, #3397, #12059, #130	1	25	31

Table 4: Steering configurations for different models. For each model, we report the index of the steered reasoning-related feature, the decoding step at which steering is applied, the steering strength α , and the steering layer.

through prompt wording. This design allows us to attribute observed differences in behavior to latent steering and prompting conditions rather than to prompt-specific artifacts.

A.4 SAE Training Information

For the Qwen3 models, we train sparse autoencoders (SAEs) from scratch using the `sparsify` library with its default configuration. All training hyperparameters follow the library defaults unless otherwise specified.

We use the EleutherAI/SmolLM2-135M-10B dataset as training data for the SAEs. This dataset is a sampled subset of the SmolLM2-135M pre-training corpus (derived from the data used to train SmolLM2 as described in (Ben Allal et al., 2025)), intended for efficient training of auxiliary components. The dataset is tokenized using the corresponding model tokenizer and processed into fixed-length chunks following the standard `chunk_and_tokenize` procedure provided by the library.

SAEs are trained on the frozen base language model in an inference-only setting. We use a batch size of 16 and train the autoencoders using the default optimization and sparsity settings. No model-specific tuning is performed beyond selecting the target layers corresponding to the Qwen3 architectures. For Qwen3-4B and Qwen3-0.6B, we train sparse autoencoders with latent dimensions of 26,624 and $32 \times$ the model hidden size, respectively. Both models use Top- K sparsity with $K = 192$.

A.5 Steering Settings

Table 4 summarizes the steering configurations used across different models. For each model, we report the index of the reasoning-related feature selected for steering, the decoding step at which the intervention is applied, and the steering strength α . These settings are held fixed across tasks and datasets for a given model, and are used consistently in all steering experiments reported in the paper.

A.6 Model-specific Configurations

All experiments use pretrained causal language models loaded from their official HuggingFace checkpoints, without any additional finetuning. Models are evaluated in inference mode with parameters frozen. We enable the output of hidden states (`output_hidden_states=True`) for all models to support activation-level analysis. Models are loaded using standard half-precision (FP16), except for Gemma models which are evaluated in full precision (FP32). Tokenizers use left padding, with the end-of-sequence token assigned as the padding token when necessary.

A.7 Point-biserial Correlation Details

We measure the association between the activation magnitude of a latent feature and a binary target variable using the point-biserial correlation. For each sample i , let x_i denote the feature activation value (in our case, the maximum activation within the first 20 decoding steps), and let $y_i \in \{0, 1\}$ denote the binary label indicating whether the sample belongs to condition A ($y_i = 1$) or condition B ($y_i = 0$).

Let n_1 be the number of samples with $y_i = 1$ and n_0 the number of samples with $y_i = 0$, with $n = n_1 + n_0$. Define the group means

$$\bar{x}_1 = \frac{1}{n_1} \sum_{i:y_i=1} x_i, \quad \bar{x}_0 = \frac{1}{n_0} \sum_{i:y_i=0} x_i,$$

and the overall sample standard deviation of x

$$s_x = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2}, \quad \text{where } \bar{x} = \frac{1}{n} \sum_{i=1}^n x_i.$$

The point-biserial correlation coefficient is then

$$r_{pb} = \frac{\bar{x}_1 - \bar{x}_0}{s_x} \sqrt{\frac{n_1 n_0}{n^2}}.$$

(Equivalently, r_{pb} is identical to the Pearson correlation between x_i and the binary variable y_i when y_i is coded as 0/1.)

To test significance under the null hypothesis $H_0 : r_{pb} = 0$, we use the standard t -test for correlation:

$$t = r_{pb} \sqrt{\frac{n-2}{1-r_{pb}^2}},$$

with degrees of freedom $n-2$. We report two-sided p -values computed from the t distribution:

$$p = 2(1 - F_{t(n-2)}(|t|)),$$

where $F_{t(n-2)}(\cdot)$ denotes the CDF of the t distribution with $n-2$ degrees of freedom.