

I Have Covered All the Bases Here: Interpreting Reasoning Features in Large Language Models via Sparse Autoencoders

Anonymous ACL submission

Abstract

Recent LLMs like DEEPSEEK-R1 have demonstrated state-of-the-art performance by integrating deep thinking and complex reasoning during generation. However, the internal mechanisms behind these reasoning processes remain unexplored. We observe *reasoning* LLMs consistently use vocabulary associated with human reasoning processes. We hypothesize these words correspond to specific reasoning moments within the models' internal mechanisms. To test this hypothesis, we employ Sparse Autoencoders (SAEs), a technique for sparse decomposition of neural network activations into human-interpretable features. We introduce *ReasonScore*, an automatic metric to identify active SAE features during these reasoning moments. We perform manual and automatic interpretation of the features detected by our metric, and find those with activation patterns matching uncertainty, exploratory thinking, and reflection. Through steering experiments, we demonstrate that amplifying these features increases performance on reasoning-intensive benchmarks (+13.4%) while producing longer reasoning traces (+20.5%). Using the model diffing technique, we provide evidence that these features are present only in models with reasoning capabilities. Our work provides the first step towards a mechanistic understanding of reasoning in LLMs.¹

1 Introduction

Large Language Models (LLMs) have achieved remarkable success in natural language processing (Brown et al., 2020), evolving beyond simple token prediction tasks towards explicit reasoning behaviors, such as step-by-step problem-solving (Wei et al., 2022; Kojima et al., 2022; Wang et al., 2022) and self-reflection (Madaan et al., 2023; Shinn et al., 2023). Recently, specialized models which we denote as *reasoning* models, such

¹Code available at <https://anonymous.4open.science/r/emnlp2025-code-anon-7E82>

Problem: Convert the point (0,3) in rectangular coordinates to polar coordinates. Enter your answer in the form (r, θ) , where $r > 0$ and $0 \leq \theta < 2\pi$.

Generation (no steering): Okay, so I have this problem where I need to convert the rectangular coordinates (0, 3) to polar coordinates.

{... 1500 tokens ...} ← **Reasoning trace**
So, putting it all together, (0, 3) in rectangular coordinates is $(3, \pi/2)$ in polar coordinates.

Generation (steering): Okay, so I have this problem here where I need to convert the point (0, 3) from rectangular (which is the same as Cartesian) coordinates to polar coordinates.

{... 2000 tokens ...} ← **Increased reasoning trace**
I think I've covered all the bases here.

Calculated r , determined θ , checked using different methods, and even considered the quadrant placement. I don't see any issues with the reasoning. So, I feel confident that the polar coordinates for the point (0, 3) are $(3, \pi/2)$.

Figure 1: Illustration of steering (amplifying) reasoning-specific features during LLM generation. Default generation (blue) shows standard model reasoning, whereas steering (green) induces increased reasoning, self-correction, and graceful transition to the final answer—evidence that the identified features are responsible for the *reasoning* concept.

as OpenAI's o1 (OpenAI, 2024b) and DEEPSEEK-R1 (Guo et al., 2025), have significantly improved performance on complex reasoning tasks. Trained through advanced fine-tuning and reinforcement learning (Shao et al., 2024), these models incorporate reasoning and reflective problem-solving by generating long chains of thought before providing final answers. These advances raise a new research question: How are such reasoning capabilities internally encoded within LLMs?

A growing body of work suggests that LLMs represent meaningful concepts as linear directions in their activation spaces (Mikolov et al., 2013; Elhage et al., 2022; Park et al., 2023; Nanda et al.,

2023; Jiang et al., 2024). However, identifying these directions remains challenging. Sparse Autoencoders (SAEs) offer a principled approach to disentangle activations into sparse, interpretable features (Cunningham et al., 2023; Gao et al., 2024b; Templeton, 2024; Marks et al., 2024). Given a trained SAE, the interpretation of its features could be performed by activation analysis (Bricken et al., 2023), targeted interventions (Templeton, 2024), or automated methods (Paulo et al., 2024; Kuznetsov et al., 2025). While SAEs have proven effective in discovering features for various concepts (Shu et al., 2025), their ability to isolate reasoning-specific features remains unexplored.

In this work, we investigate whether reasoning processes in *reasoning* LLMs can be identified and decomposed into interpretable directions within their activation spaces. We analyze the outputs produced by these models’, and find a consistent pattern in which they employ words associated with human reasoning processes: uncertainty (e.g. “perhaps”), reflection (e.g. “however”), and exploration (e.g. “alternatively”) (Chinn and Anderson, 1998; Boyd and Kong, 2017; Gerns and Mortimore, 2025). We hypothesize that these linguistic patterns correspond to the moments of reasoning within the models’ internal mechanisms. To test this, we construct a vocabulary of reasoning words. We then use SAEs to decompose LLM activations into interpretable features and propose ReasonScore, a metric that quantifies the degree to which a given SAE feature is active on the reasoning vocabulary.

We evaluate the features found by ReasonScore using manual (Bricken et al., 2023) and automatic interpretation (Kuznetsov et al., 2025) techniques, and find the set of 46 features that demonstrate interpretable activation patterns corresponding to uncertainty, exploratory thinking, and reflection. We perform steering experiments and show that amplifying these reasoning features leads to improved performance on reasoning-intensive benchmarks (+13.4% on AIME-2024, +2.2% on MATH-500, and +4% on GPQA Diamond) while producing longer reasoning traces (+18.5% on AIME-2024, +20.5% on MATH-500, and +13.9% on GPQA Diamond). Through model diffing (Bricken et al., 2024), we demonstrate that these reasoning features emerge only in *reasoning* LLMs and are absent in base models. Our results provide mechanistic evidence that specific, interpretable components in LLMs representations are causally linked to reasoning behavior.

The contributions of this paper are the following:

- We introduce ReasonScore, an automatic metric to identify the SAE features responsible for reasoning and confirm its effectiveness using interpretability techniques.
- We provide causal evidence from steering experiments, demonstrating that amplifying identified features induces reasoning behavior.
- We analyze the emergence of reasoning features in LLMs through model diffing technique, and confirm their existence only after the *reasoning* fine-tuning stage.

2 Interpretability with SAEs

Sparse Autoencoders (SAEs) aim to learn a sparse decomposition of model activations to identify disentangled features that correspond to meaningful concepts (Bricken et al., 2023). Here, a *feature* refers to an individual component of the learned representation that captures specific, human-interpretable characteristics of the input data.

The core idea behind SAEs is to reconstruct model activations $x \in \mathbb{R}^n$ as a sparse linear combination of learned feature directions, where the feature *dictionary* dimensionality $m \gg n$. Formally, we extract LLM activations from some intermediate state in the model and train a two-layer autoencoder:

$$\begin{aligned} f(x) &= \sigma(W_{\text{enc}}x + b_{\text{enc}}), \\ \hat{x}(f) &= W_{\text{dec}}f + b_{\text{dec}}. \end{aligned} \quad (1)$$

Here, $f(x) \in \mathbb{R}^m$ is a sparse vector of feature magnitudes and $\hat{x}(f) \in \mathbb{R}^n$ is a reconstruction of the original activation x . The columns of W_{dec} , which we denote by $W_{\text{dec},i}, i = 1, \dots, m$, represent the dictionary of directions, or *features*, into which the SAE decomposes x . The activation function σ enforces non-negativity in $f(x)$.

The training objective used to train Sparse Autoencoders minimizes a reconstruction loss $\mathcal{L}_{\text{recon}}$ and an additional sparsity-promoting loss $\mathcal{L}_{\text{sparsity}}$. This objective forces SAE to learn a small set of interpretable features that capture the distinct properties of the activations.

In our work, we use vanilla SAE (Bricken et al., 2023) with ReLU activation function. Following (Conerly et al., 2024), we use a squared error reconstruction loss and a modified L1 penalty as a

sparsity loss:

$$\mathcal{L} = \underbrace{\|x - \hat{x}\|_2^2}_{\mathcal{L}_{\text{recon}}} + \lambda \underbrace{\sum_{i=1}^m f_i \|W_{\text{dec},i}\|_2}_{\mathcal{L}_{\text{sparsity}}}, \quad (2)$$

where λ is the sparsity penalty coefficient.

3 Method

We identify reasoning-specific features through a two-step approach. First, we examine the language space of reasoning words used by *reasoning* LLMs, and construct the respective vocabulary \mathcal{R} (Sec. 3.1). Secondly, we introduce ReasonScore to find the sparse autoencoder features responsible for reasoning capabilities (Sec. 3.2).

3.1 Reasoning Vocabulary

Reasoning words are linguistic features associated with exploratory talk as humans talk-to-learn, explore ideas, and probe each other’s thinking (Boyd and Kong, 2017).

In the original DEEPSEEK-R1 paper (Guo et al., 2025), the authors demonstrated that the model spontaneously exhibits sophisticated human-like behaviors, such as reflection, where it revisits and reevaluates its previous steps, and exploration of alternative problem-solving approaches. In particular, the model explicitly employs words that mirror the introspective language humans use when thinking (such as “maybe”, “but”, “wait”). We hypothesize that these moments correspond directly to the internal reasoning process of the models, which is consistent with studies on human thinking (Chinn and Anderson, 1998; Boyd and Kong, 2017).

To extract the models’ reasoning vocabulary, we use an approach similar to that of (Rayson and Garside, 2000). We construct two corpora from the OPENTHOUGHTS-114K (OpenThoughts, 2025) dataset: ground-truth samples containing formal and step-by-step solutions to the problems, and the solutions obtained using DEEPSEEK-R1 for the same problems. For each word, we calculate its frequency in the tasks solutions p_{sol} , and in the thinking solutions p_{thinks} . Then, we sort all the words by the difference $p_{\text{thinks}} - p_{\text{sol}}$ and filter out those with a high presence in the Google Books Ngram Corpus (Michel et al., 2011). As the filtering threshold, we choose the 99.9995-th percentile of all n-grams probabilities from the Ngram Corpus. Figure 2 shows the distribution of the top 40 words. We then select 10 words indicating reasoning as models’ *reasoning* vocabulary and denote

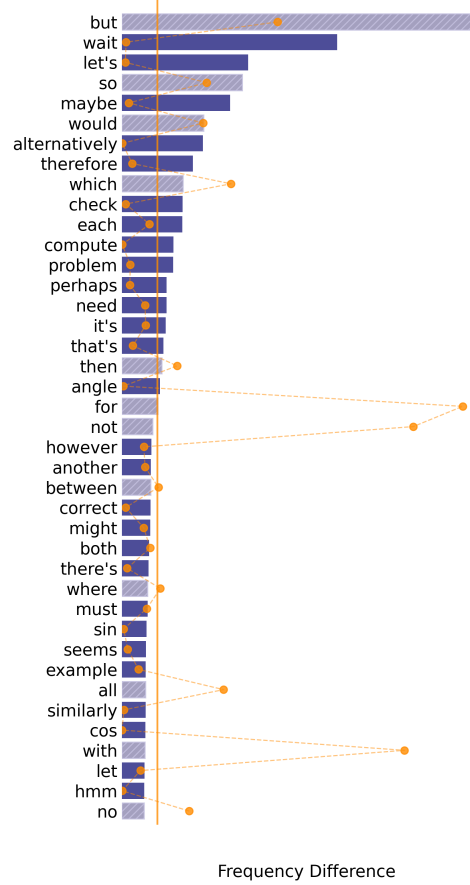


Figure 2: The distribution of top 40 words with the greatest change in frequency between reasoning traces of DEEPSEEK-R1 and ground-truth solutions of math problems. Orange dots show the frequency from Google Books Ngram Corpus. We remove the words with absolute frequency above the pre-defined threshold (orange line), and keep those with the high relative frequency indicating reasoning.

it by \mathcal{R} . The exact list of words can be found in Appx. A.1.

3.2 ReasonScore

To find SAE features that capture reasoning-related behavior, we follow our hypothesis and introduce ReasonScore, which measures the contribution of i -th feature to reasoning. Using a dataset of model’s activations (see details in Sec. 4.1) $\mathcal{D} = \mathcal{D}_{\mathcal{R}} \cup \mathcal{D}_{\neg\mathcal{R}}$, where $\mathcal{D}_{\mathcal{R}}$ contains token activations corresponding to words in \mathcal{R} and $\neg\mathcal{D}_{\mathcal{R}}$ contains all other activations, we first define a score:

$$s_i = \frac{\mu(i, \mathcal{D}_{\mathcal{R}})}{\sum_j \mu(j, \mathcal{D}_{\mathcal{R}})} - \frac{\mu(i, \neg\mathcal{D}_{\mathcal{R}})}{\sum_j \mu(j, \neg\mathcal{D}_{\mathcal{R}})}, \quad (3)$$

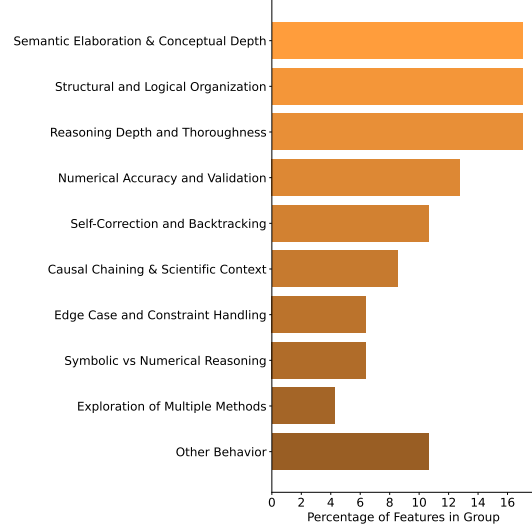
where $\mu(i, \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} f_i(x)$ is the average activation value of the i -th feature on dataset \mathcal{D} . This

Feature #4395
 But how to verify this. Let me re-express the inequalities hold. But wait, maybe the constant c up to some equivalence. But let me check if there So this is unclear. But looking at the sample

Feature #25953
 rahedra. Wait, here's another idea. (71 pm). Wait, maybe different sources have slightly formula has a different sign. Alternatively, maybe the coefficient low. Let me verify. Another way: I remember

Feature #46691
 mass ranges. Hmm. Now, detection. LIGO the starting points. That seems a bit counterintuitive trigonometric identities. Hmm, but maybe a synthetic regardless of starting values. Interesting. Let me try to

Feature #61104
 some code?), but that's not standard. Alternatively, is in a different row. That doesn't make sense That's a stretch, but perhaps that's the connection then maybe more. Wait, this is getting complicated



(a) Top-activating examples from the manually verified set of features. (b) Distribution of manually verified set of features on function groups generated by GPT-4O.

Figure 3: Interpretability results for manually verified set of features in our SAE: (a) Examples of feature interfaces used in manual interpretation experiments, (b) Distribution of reasoning features on function groups obtained by automatic interpretation pipeline by using GPT-4O as a judge.

score is similar to the one in (Cywiński and Deja, 2025) and identifies features that concentrate the most of their activation mass on reasoning words.

However, analysis of feature activations only on individual words may miss important contextual information. The words in \mathcal{R} are critical indicators of the reasoning process and also serve as transition points, signaling shifts in the thought process, uncertainty, or reflection. Therefore, a feature involved in reasoning should activate not only on the reasoning words, but also as the model approaches and continues through these transitions. To capture it, we define $\mathcal{D}_{\mathcal{R}}^{\mathbf{W}}$ as the dataset that contains activations within a fixed-width context window around tokens corresponding to words in \mathcal{R} , and $\mathcal{D}_{-\mathcal{R}}^{\mathbf{W}}$ contains all other activations. We modify Eq. 3 to use the new version of the datasets.

To penalize features that activate only on a small fraction of \mathcal{R} , we further introduce an *entropy penalty*. For i -th feature, we first calculate $\mu(i, \mathcal{D}_{r_j}^{\mathbf{W}})$ for each word $r_j \in \mathcal{R}$, normalize these values into a probability distribution $p_i(r_j) = \frac{\mu(i, \mathcal{D}_{r_j}^{\mathbf{W}})}{\sum_{k \in \mathcal{R}} \mu(i, \mathcal{D}_{r_k}^{\mathbf{W}})}$, and compute the entropy:

$$H_i = -\frac{1}{\log |\mathcal{R}|} \cdot \sum_j p_i(r_j) \log p_i(r_j). \quad (4)$$

Here, $\log |\mathcal{R}|$ normalizes the entropy to $[0, 1]$, with

$H_i = 1$ indicating perfect uniformity over \mathcal{R} . By adding the entropy penalty in Eq. (3), we define the ReasonScore for the i -th SAE feature as:

$$\text{ReasonScore}_i = \frac{\mu(i, \mathcal{D}_{\mathcal{R}}^{\mathbf{W}})}{\sum_j \mu(j, \mathcal{D}_{\mathcal{R}}^{\mathbf{W}})} \cdot H_i^\alpha - \frac{\mu(i, \mathcal{D}_{-\mathcal{R}}^{\mathbf{W}})}{\sum_j \mu(j, \mathcal{D}_{-\mathcal{R}}^{\mathbf{W}})}. \quad (5)$$

where α controls the trade-off between specificity ($\alpha \rightarrow 0$) and generalization ($\alpha > 1$).

We identify the set of reasoning features in a SAE based on their ReasonScore and define the corresponding set of feature indices as:

$$\mathcal{F}_{\mathcal{R}} = \{i \mid i \in [1, m], \text{ReasonScore}_i > \tau\} \quad (6)$$

where τ is the q -th quantile of the ReasonScore distribution across all features.

4 Evaluation

In this section, we analyze how effectively our discovered features model reflection, uncertainty, and exploration within the *reasoning* model. We discuss our experimental setup (Sec. 4.1), perform manual and automatic interpretation of the features we find (Sec. 4.2), and conduct steering experiments with these features on various benchmarks (Sec. 4.3). Finally, we apply the model diffing technique to demonstrate that these features exist only in models with reasoning capabilities (Sec. 4.4).

4.1 Experimental Setup

Model. We apply SAE to the output activations from the 19-th layer of the DEEPSEEK-R1-LLAMA-8B model. This model was selected for its *reasoning* capabilities and open-source availability. The 19-th layer ($\approx 60\%$ model depth) was chosen because at this point LLMs predominately store the most of their knowledge (Chen et al., 2023; Jin et al., 2024).

Data. We train SAE on the activations of the model generated using text data from the LMSYS-CHAT-1M (Zheng et al., 2023) and OPENTHOUGHTS-114K (OpenThoughts, 2025) datasets. The first provides a broad and diverse spectrum of real-world text data, which we denote as *base data*, while the latter provides high-quality reasoning traces generated by DEEPSEEK-R1 for math, science, code, and puzzle samples, which we denote as *reasoning data*. The SAE is trained on 1B tokens, evenly split between the two datasets, with a context window of 1,024 tokens.

Training. We set the SAE dictionary dimensionality to $m = 65,536$, which is 16 times larger than the model activation size $n = 4,096$ following established practices (Lieberum et al., 2024), and adopt the same training settings as in the Anthropic April update (Conerly et al., 2024). We train with the Adam optimizer (Kingma and Ba, 2014) with $(\beta_1, \beta_2) = (0.9, 0.999)$, batch size of 4,096, and a learning rate $\eta = 5 \times 10^{-5}$. The learning rate is decayed linearly to zero over the last 20% of training. The gradient norm is clipped to 1. We use a linear warmup for the sparsity coefficient from $\lambda = 0$ to $\lambda = 5$ over the first 5% training steps.

Evaluation. We use the mean L0-norm of latent activations, $\mathbb{E}_x \|f(x)\|_0$, as a measure of sparsity. To measure reconstruction quality, we use fraction of variance of the input explained by the reconstruction. Both metrics were computed on 2,048 sequences of length 1,024.

At a L0 of 86 the reconstruction of our SAE explains 68.5% of the variance in model activations. This shows that our SAE achieves reliable reconstruction performance at a low sparsity level, allowing a decomposition of raw activations into interpretable features.

ReasonScore. We calculate ReasonScore (Eq. 5) on 10M tokens from the OPENTHOUGHTS-114K dataset. To collect $\mathcal{D}_{\mathcal{R}}^W$, we use an

Feature #	AIME 2024		MATH-500		GPQA Diamond	
	pass@4	tokens (K)	pass@4	tokens (K)	pass@4	tokens (K)
No steering	53.3	12.4	93.2	3.9	50.0	7.9
3942	56.7	11.1	93.0	3.4	46.5	6.7
4395	56.7	14.7	95.4	4.1	52.0	8.5
16441	60.0	14.0	95.0	4.1	54.0	8.3
16778	56.7	14.1	94.0	4.7	51.0	9.0
25953	60.0	12.8	94.2	4.2	53.0	8.1
46691	56.7	14.0	94.2	4.2	54.0	8.0
61104	66.7	12.0	95.0	3.6	53.0	7.5

Table 1: Performance and average number of output tokens for different steering experiments on reasoning-related benchmarks.

asymmetric window with 2 preceding and 3 following tokens. We set $\alpha = 0.7$ for the *entropy penalty* as a reasonable default. Based on the empirical analysis of ReasonScore distribution (see Appx. A.2), we set $q = 0.997$ in Eq. (6), resulting in $|\mathcal{F}_{\mathcal{R}}| = 200$ features.

4.2 Interpretability of Reasoning Features

Manual Interpretation. To evaluate the features we found with ReasonScore, we manually interpret each feature from $\mathcal{F}_{\mathcal{R}}$ (200 in total). For each feature, we find the examples in a subset of the OPENTHOUGHTS-114K corpus that caused the feature to activate, and construct the interface proposed in (Bricken et al., 2023). This mainly includes examples of when the feature activates, its effect on the logits when it does, and other statistics. We determine whether a feature qualifies as a good reasoning candidate if: (1) when it is active, the relevant concept is reliably present in the context, (2) it triggers in various examples of reasoning tasks, and (3) its activation impacts interpretable logits that correspond to reasoning processes.

Our manual analysis reveals a set of 46 features, which we denote as $\mathcal{F}_{\mathcal{R}}^{\text{manual}} \subset \mathcal{F}_{\mathcal{R}}$, that we believe to be responsible for the reasoning mechanisms of the model. In Fig. 3a, we provide examples of feature interfaces used for interpretation. The results demonstrate features that consistently activate in contexts representing model’s uncertainty (#61104), exploration (#25953), and reflection (#4395, #46691). Additional examples of interfaces can be found in Appx. B.1.

Automatic Interpretation. To complement our manual analysis, we annotate these features with an automatic interpretation pipeline (Kuznetsov et al., 2025). This approach employs feature steering, a technique that modulates feature activations to analyze their functional influence. For each i -th

feature, we estimate its maximum activation f_i^{\max} using a subset of the OPENTHOUGHTS-114K corpus. During response generation, we modify model activations as follows:

$$x' = x + \gamma f_i^{\max} W_{\text{dec},i}, \quad (7)$$

where γ controls the steering strength.

To evaluate the impact of i -th feature on reasoning capabilities, we generate multiple outputs by varying $\alpha \in [-4, 4]$, pass them to GPT-4O, and ask it to generate an explanation or function that best describes the semantic influence caused by steering a feature. The result, shown in Fig. 3b, reveals that the features we found group into distinct reasoning-related patterns. Only a small fraction of features from $\mathcal{F}_{\mathcal{R}}^{\text{manual}}$ (5) was assigned to a mixed class ‘‘Other Behavior’’ containing mixed explanation. We provide a more comprehensive description of auto-interpretability pipeline results in Appx. B.2.

Takeaway 1: Manual interpretation experiments confirm that ReasonScore identifies features that describe model’s reasoning capabilities, revealing 46 features that represent uncertainty, exploration, and reflection. Automatic interpretation demonstrates that these features are causally linked to reasoning behavior.

4.3 Steering Reasoning Features

To demonstrate whether our interpretations of features describe their influence on model behavior, we further experiment with feature steering.

Our goal is to verify if steering reasoning features improve the LLM’s performance on reasoning-related benchmarks. Following the setup in DEEPSEEK-R1, we evaluate performance on AIME 2024 (of America, 2024), MATH-500 (Hendrycks et al., 2021), and GPQA Diamond (Rein et al., 2023). To obtain steering results for i -th feature, we modify the activations during response generation according to Eq. (7). To determine the optimal steering strength that can influence model outputs without significantly damaging capabilities, we ran evaluations with a small subset of 10 reasoning features on MATH-500. We varied the steering strength γ from 1 to 8. Based on these experiments, we determined the optimal range $\gamma \in [1, 3]$, which aligns with the findings in

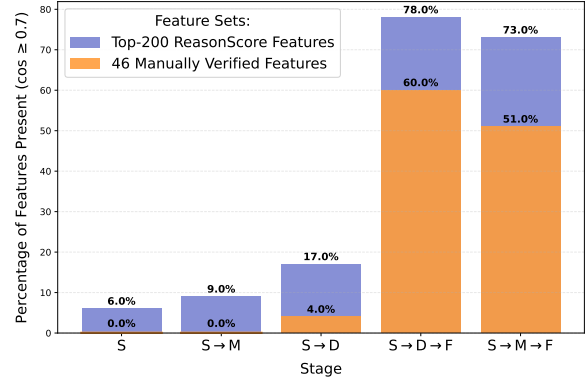


Figure 4: Percentage of ReasonScore features present at each stage of the diffing pipeline. The blue bars represent the features from $\mathcal{F}_{\mathcal{R}}$, the orange bars represent the $\mathcal{F}_{\mathcal{R}}^{\text{manual}}$ features. Features are considered present if their cosine similarity with any feature in corresponding stage’s SAE is ≥ 0.7 . Stages: (S) Base model + base data; (S→D) Base model + reasoning data; S→M Reasoning model + base data; (S→D/M→F) Reasoning model + reasoning data.

(Durmus et al., 2024). For all subsequent experiments, we set the steering strength $\gamma = 2$.

We perform a preliminary analysis to identify the most promising features for reasoning enhancement from our set of manually chosen features $\mathcal{F}_{\mathcal{R}}^{\text{manual}}$. For each feature, we measure the accuracy (or pass@1 (Chen et al., 2021)) on MATH-500 and evaluate the results. Of the 46 features, 9 improve performance by $\geq 0.5\%$, 29 show no or minimal performance degradation (≤ 2.0), and the remaining 8 decrease performance by at most 4%. Interestingly, we identify feature #3942, which produces substantially shorter responses while maintaining negligible performance degradation. For further analysis, we select the 9 top-performing features and feature #3942.

We evaluate these 10 features across all reasoning benchmarks. We report pass@4 and the average number of tokens generated during the model’s thinking process. The results, shown in Tab. 1, demonstrate that steering 7 out of 10 features produces consistent improvements in both performance and reasoning depth. Feature #61104 yields the most significant performance gain on AIME-2024 (+13.4%). Feature #16778 produces the longest reasoning traces on average (+13.7% on AIME-2024, +20.5% on MATH-500, and +13.9% on GPQA Diamond) and consistently outperforms the ‘‘no steering’’ baseline. Feature #3942 produces shortest reasoning traces on average (−7.7%) with minor performance degradation.

We provide examples of generated solutions without and with feature steering in Appx. C.

Takeaway 2: We find that amplifying certain reasoning features prolongs the internal thought process and correlates with increased performance on reasoning-related tasks.

4.4 Stage-wise Emergence of Reasoning Features

Our interpretation experiments (Sec. 4.2) revealed that features identified by ReasonScore exhibit activation patterns consistent with reasoning processes. The steering experiments (Sec. 4.3) provided causal evidence by demonstrating that amplification of these features improves performance on reasoning-intensive benchmarks. Given these findings, we now aim to answer the next important question: do these reasoning features naturally emerge during standard pre-training procedure, or are they specifically induced by the *reasoning* fine-tuning process?

To answer this question, we use the stage-wise fine-tuning technique proposed in (Bricken et al., 2024). This approach aims to isolate how features evolve across different model and dataset combinations. In our experiments, we examine how the features change between two model states: before (*base model*) and after (*reasoning model*) *reasoning* fine-tuning stage. We accomplish this by training a SAE on the base model before it has been fine-tuned, and then fine-tuning it on either the reasoning model or the fine-tuning data. Formally, we define four distinct stages:

Stage S: base model + base data (starting point)

Stage D: base model + reasoning data (isolating dataset effects)

Stage M: reasoning model + base data (isolating model effects)

Stage F: reasoning model + reasoning data (full fine-tuning)

We analyze these changes through two fine-tuning trajectories, each involving two sequential fine-tuning stages: (1) $S \rightarrow D \rightarrow F$ takes initial SAE (Stage S), fine-tunes it on reasoning data ($S \rightarrow D$), and finally fine-tunes on both reasoning model and reasoning data ($D \rightarrow F$); (2) $S \rightarrow M \rightarrow F$ takes initial

SAE (Stage S), fine-tunes it on reasoning model ($S \rightarrow M$), and finally fine-tunes on both reasoning model and reasoning data ($M \rightarrow F$). If reasoning features are present only in *reasoning* models, we should observe the emergence of these features in response to **both** reasoning model and reasoning data (Stage F). This corresponds to the final steps of the fine-tuning trajectories: ($S \rightarrow D/M \rightarrow F$).

We use LLAMA-3.1-8B (Grattafiori et al., 2024) as base model and SLIMPAJAMA (Soboleva et al., 2023) as base data. We select SLIMPAJAMA over LMSYS-CHAT-1M as our *base data* because it better matches the pre-training distribution of LLAMA-3.1-8B, which has not undergone instruction-tuning. For each stage, we use the same setup as in Sec. 4.1, with each fine-tuning stage taking 30% of the total number of tokens required for training from scratch. For each i -th feature from \mathcal{F}_R , we check its existence at each stage by measuring cosine similarity (\cos) between feature vectors. We follow (Bricken et al., 2024) and consider a feature present if $\cos \geq 0.7$ with any feature in a SAE of the corresponding stage.

Fig. 4 shows the percentage of reasoning features present at each fine-tuning stage. We find that the reasoning features are almost absent in the base model and after switching to the reasoning model (0% of manually verified features $\mathcal{F}_R^{\text{manual}}$). When introducing the reasoning data to the base model ($S \rightarrow D$), only 4% of the verified reasoning features emerge, indicating that exposure to the reasoning content alone is insufficient to develop these features. Finally, when we incorporate both the reasoning data and the reasoning model, we observe that 60% of the verified reasoning features appear in the ($S \rightarrow D \rightarrow F$) stage and 51% in the ($S \rightarrow M \rightarrow F$) stage. The noticeable increase in the presence of features only when both reasoning data and model are combined provides compelling evidence that ReasonScore identifies features associated with the model’s reasoning processes rather than general capabilities.

Takeaway 3: We show that most of the features found by ReasonScore emerge only after the *reasoning* fine-tuning stage. Exposure to the reasoning data or reasoning model alone is insufficient to develop these features.

5 Related Work

5.1 Mechanistic Interpretability

Various methods exist to shed light on the inner workings of LLMs, including attention analysis, which examines the model’s focus on input tokens (Vaswani et al., 2017), and gradient-based methods that identify influential input features (Simonyan et al., 2014). Probing techniques offer insights into the information captured within different layers of an LLM (Alain and Bengio, 2016). Mechanistic interpretability aims to reverse-engineer the computations of LLMs, employing techniques like activation patching (Meng et al., 2022) and feature steering (Cao et al., 2024; Soo et al., 2025) to understand and control model behavior. The logit lens provides a way to observe the model’s token predictions at different processing stages (Nostalgebraist, 2020).

5.2 Sparse Autoencoders

Sparse Autoencoders (SAEs) have emerged as a key tool for understanding the internal representations of LLMs, particularly in the context of interpretability research (Gao et al., 2024a; Huben et al., 2024). By learning a sparse decomposition of model activations, SAEs can identify disentangled features that correspond to meaningful concepts (Marks et al., 2024).

Previous work has shown that SAE features are significantly more monosemantic than individual neurons, making them an effective tool for mechanistic interpretability (Leask et al., 2025). A key challenge in using SAEs for interpretability is ensuring extracted features are monosemantic and robust. Yan et al. (Yan et al., 2024) propose using feature decorrelation losses to enforce better separation between learned latents, preventing redundancy in feature representations. Furthermore, recent advances in cross-layer SAEs (Shi et al., 2025) have allowed more abstract, high-level reasoning patterns by analyzing activations across multiple transformer layers.

SAEs have also proven valuable for studying model development across training stages. Cross-coders (Lindsey et al., 2024) enable direct mapping of features to model states, while stage-wise model diffing (Bricken et al., 2024) compares SAEs trained on different checkpoints. In our experiments, we adopt the diffing approach for its computational efficiency and intuitive implementation. While previous work has applied diffing to sleeper

agents, our research extends this approach to investigate reasoning behavior.

5.3 Reasoning LLMs

Recent innovations in LLMs have shifted the focus toward models with explicit reasoning abilities. Notable examples include OpenAI’s o1 (OpenAI, 2024b), DEEPSEEK-R1 (Guo et al., 2025) and QWQ-32B-PREVIEW (Team, 2024). These methods train LLMs by rule-based reinforcement learning that employs the correctness score (whether the final answer is correct or not) and a format score (to ensure outputs follow a predetermined structure) which leads to the emergence of advanced reasoning behaviors like self-correction and reflection, denoted as an “aha moment” in the DEEPSEEK-AI (Guo et al., 2025) report.

Despite the success of rule-based reinforcement learning and iterative fine-tuning in enabling reasoning capabilities, how these models encode their internal reasoning remains an open question. In our work, we study this problem by using a Sparse Autoencoder to find interpretable features responsible for the underlying reasoning mechanisms inside these LLMs, which, to the best of our knowledge, has not been done yet.

6 Conclusion

In this work, we present a novel methodology for uncovering the internal mechanisms of reasoning in LLMs through the lens of Sparse Autoencoders. We introduce ReasonScore, a metric that identifies reasoning-related SAE features based on their activation patterns using a curated introspective vocabulary. Our interpretation experiments reveal a subset of these features that demonstrate interpretable activation patterns corresponding to diverse reasoning behaviors: uncertainty, exploratory thinking, and self-reflection. Through steering experiments, we provide causal evidence that certain features selected by ReasonScore directly correspond to the model’s reasoning behaviors. Amplifying them prolongs the internal thought process and increases performance across multiple reasoning-related benchmarks. Using a stage-wise fine-tuning technique, we confirm that most of the features found by ReasonScore emerge only after the *reasoning* fine-tuning process. Our work provides the first mechanistic evidence that specific, interpretable components of LLM representations are causally linked to complex reasoning behaviors.

Limitations

ReasonScore While ReasonScore has proven to identify reasoning-specific features, it has several limitations. Our metric depends on multiple hyperparameters, such as context window size and entropy penalty coefficient α . Although we obtained good results with the default parameters, ablation studies should be performed to determine the optimal values. Our feature selection process yielded 46 interpretable features from the top 200 candidates (23%). While other features can also contribute to reasoning, we could not confidently classify them due to ambiguous activation patterns. Among the unverified features, we also observed several token-level features that activate on reasoning words but lack clear conceptual meaning. Finally, our reasoning vocabulary might not comprehensively capture all patterns associated with diverse forms of reasoning in LLMs. These limitations suggest opportunities for refinement in future work.

Sparse Autoencoders SAEs provide a powerful framework for interpreting LLMs. However, it is known that SAEs suffer from some problems that complicate the extraction of fully interpretable features (Chanin et al., 2024; Leask et al., 2025). While we have found multiple interpretable features with causal influence, we could have also missed other interesting features. We also restricted our analysis to layer 19 of DEEPSEEK-R1-LLAMA-8B model, potentially missing reasoning mechanisms distributed across other layers. These limitations highlight both the strengths and current constraints of using SAEs for mechanistic interpretability.

Emergence of Reasoning Features While the results in Sec. 4.4 support our hypothesis, we acknowledge certain limitations of the diffing approach. First, we choose the cosine similarity threshold (0.7) empirically following the initial work, which might miss similar features if the representation is rotated during one of the fine-tuning stages. A single feature can also be split (or absorbed) and does not cross the defined threshold. Second, we identify only 60% of the manually verified reasoning features (and 78% of \mathcal{F}_R features) in the final stage, with the remaining features missing. These incompleteness probably results from fine-tuning the SAE rather than training from scratch at each stage, and the instability of SAE decom-

position in preserving features between different runs (Paulo and Belrose, 2025). These limitations show that our approach can result in false negative and false positive predictions. However, we believe that our primary finding remains valid even under these limitations.

Ethics Statement

Use of AI Assistants We utilize Grammarly to enhance and proofread the text of this paper, correcting grammatical, spelling, and stylistic errors, as well as rephrasing sentences. Consequently, certain sections of our publication may be identified as AI-generated, AI-edited, or a combination of human and AI contributions.

References

- Guillaume Alain and Yoshua Bengio. 2016. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*.
- Maureen Boyd and Yiren Kong. 2017. Reasoning words as linguistic features of exploratory talk: Classroom use and what it can tell us. *Discourse Processes*, 54(1):62–81.
- Trenton Bricken, Siddharth Mishra-Sharma, Jonathan Marcus, Adam Jermy, Christopher Olah, Kelley Rivoire, and Thomas Henighan. 2024. Stage-wise model diffing. *Transformer Circuits Thread*.
- Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermy, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, and 1 others. 2023. Towards monosemanticity: Decomposing language models with dictionary learning, 2023. URL <https://transformer-circuits.pub/2023/monosemantic-features/index.html>, page 9.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.
- Yuanpu Cao, Tianrong Zhang, Bochuan Cao, Ziyi Yin, Lu Lin, Fenglong Ma, and Jinghui Chen. 2024. Personalized steering of large language models: Versatile steering vectors through bi-directional preference optimization. *arXiv preprint arXiv:2406.00045*.
- David Chanin, James Wilken-Smith, Tomáš Dulka, Hardik Bhatnagar, and Joseph Bloom. 2024. A is for absorption: Studying feature splitting and absorption in sparse autoencoders. *arXiv preprint arXiv:2409.14507*.

705	Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan,	Daya Guo, Dejian Yang, Haowei Zhang, Junxiao	758
706	Henrique Ponde De Oliveira Pinto, Jared Kaplan,	Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shi-	759
707	Harri Edwards, Yuri Burda, Nicholas Joseph, Greg	rong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025.	760
708	Brockman, and 1 others. 2021. Evaluating large	Deepseek-r1: Incentivizing reasoning capability in	761
709	language models trained on code. <i>arXiv preprint</i>	llms via reinforcement learning. <i>arXiv preprint</i>	762
710	<i>arXiv:2107.03374</i> .	<i>arXiv:2501.12948</i> .	763
711	Nuo Chen, Ning Wu, Shining Liang, Ming Gong, Linjun	Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul	764
712	Shou, Dongmei Zhang, and Jia Li. 2023. Beyond sur-	Arora, Steven Basart, Eric Tang, Dawn Song, and	765
713	face: Probing llama across scales and layers. <i>arXiv</i>	Jacob Steinhardt. 2021. Measuring mathematical	766
714	<i>preprint arXiv:2312.04333</i> .	problem solving with the math dataset . <i>Preprint</i> ,	767
715	Clark A. Chinn and Richard C. Anderson. 1998. The	arXiv:2103.03874.	768
716	structure of discussions that promote reasoning.	Robert Huben, Hoagy Cunningham, Logan R. Smith,	769
717	<i>Teachers College Record</i> , 100(2):315–368.	Aidan Ewart, and Lee Sharkey. 2024. Sparse autoen-	770
718	Tom Conerly, Adly Templeton, Trenton Bricken,	coders find highly interpretable features in language	771
719	Jonathan Marcus, and Tom Henighan. 2024. Update	models. In <i>Proceedings of the International Confer-</i>	772
720	on how we train saes. <i>Transformer Circuits Thread</i> .	<i>ence on Learning Representations (ICLR)</i> .	773
721	Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert	Yibo Jiang, Goutham Rajendran, Pradeep Ravikumar,	774
722	Huben, and Lee Sharkey. 2023. Sparse autoencoders	Bryon Aragam, and Victor Veitch. 2024. On the	775
723	find highly interpretable features in language models .	origins of linear representations in large language	776
724	<i>Preprint</i> , arXiv:2309.08600.	models. <i>arXiv preprint arXiv:2403.03867</i> .	777
725	Bartosz Cywiński and Kamil Deja. 2025. Saeu-	Mingyu Jin, Qinkai Yu, Jingyuan Huang, Qingcheng	778
726	ron: Interpretable concept unlearning in diffusion	Zeng, Zhenting Wang, Wenyue Hua, Haiyan Zhao,	779
727	models with sparse autoencoders. <i>arXiv preprint</i>	Kai Mei, Yanda Meng, Kaize Ding, and 1 others.	780
728	<i>arXiv:2501.18052</i> .	2024. Exploring concept depth: How large language	781
729	Esin Durmus, Alex Tamkin, Jack Clark, Jerry Wei,	models acquire knowledge and concept at different	782
730	Jonathan Marcus, Joshua Batson, Kunal Handa,	layers? <i>arXiv preprint arXiv:2404.07066</i> .	783
731	Liane Lovitt, Meg Tong, Miles McCain, and 1 others.	Diederik P Kingma and Jimmy Ba. 2014. Adam: A	784
732	2024. Evaluating feature steering: A case study in	method for stochastic optimization. <i>arXiv preprint</i>	785
733	mitigating social biases, 2024. URL https://anthropic.	<i>arXiv:1412.6980</i> .	786
734	com/research/evaluating-feature-steering .	Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yu-	787
735	Nelson Elhage, Tristan Hume, Catherine Olsson,	taka Matsuo, and Yusuke Iwasawa. 2022. Large lan-	788
736	Nicholas Schiefer, Tom Henighan, Shauna Kravec,	guage models are zero-shot reasoners. <i>Advances in</i>	789
737	Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain,	<i>neural information processing systems</i> , 35:22199–	790
738	Carol Chen, and 1 others. 2022. Toy models of su-	22213.	791
739	perposition. <i>arXiv preprint arXiv:2209.10652</i> .	Kristian Kuznetsov, Laida Kushnareva, Polina Druzhin-	792
740	Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel	ina, Anton Razzhigaev, Anastasia Voznyuk, Irina	793
741	Goh, Rajan Troll, Alec Radford, Ilya Sutskever,	Piontkovskaya, Evgeny Burnaev, and Serguei Baran-	794
742	Jan Leike, and Jeffrey Wu. 2024a. Scaling and	nikov. 2025. Feature-level insights into artificial text	795
743	evaluating sparse autoencoders. <i>arXiv preprint</i>	detection with sparse autoencoders. <i>arXiv preprint</i>	796
744	<i>arXiv:2406.04093</i> .	<i>arXiv:2503.03601</i> .	797
745	Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel	Patrick Leask, Bart Bussmann, Michael T. Pearce,	798
746	Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan	Joseph I. Bloom, Curt Tigges, Noura Al Moubayed,	799
747	Leike, and Jeffrey Wu. 2024b. Scaling and evaluating	Lee Sharkey, and Neel Nanda. 2025. Sparse autoen-	800
748	sparse autoencoders . <i>Preprint</i> , arXiv:2406.04093.	coders do not find canonical units of analysis. In <i>Pro-</i>	801
749	Pilar Gerns and Louisa Mortimore. 2025. Towards ex-	<i>ceedings of the International Conference on Learning</i>	802
750	ploratory talk in secondary-school cli: An empirical	<i>Representations (ICLR)</i> .	803
751	study of the cognitive discourse function ‘explore’.	Tom Lieberum, Senthooran Rajamanoharan, Arthur	804
752	<i>Language Teaching Research</i> .	Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant	805
753	Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri,	Varma, János Kramár, Anca Dragan, Rohin Shah,	806
754	Abhinav Pandey, Abhishek Kadian, Ahmad Al-	and Neel Nanda. 2024. Gemma scope: Open sparse	807
755	Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten,	autoencoders everywhere all at once on gemma 2.	808
756	Alex Vaughan, and 1 others. 2024. The llama 3 herd	<i>arXiv preprint arXiv:2408.05147</i> .	809
757	of models. <i>arXiv preprint arXiv:2407.21783</i> .	Jack Lindsey, Adly Templeton, Jonathan Marcus,	810
		Thomas Conerly, Joshua Batson, and Christopher	811
		Olah. 2024. Sparse crosscoders for cross-layer	812
		features and model diffing. <i>Transformer Circuits</i>	813
		<i>Thread</i> .	814

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.

Hanqi Yan, Yanzheng Xiang, Guangyi Chen, Yifei Wang, Lin Gui, and Yulan He. 2024. Encourage or inhibit monosemanticity? revisiting monosemanticity from a feature decorrelation perspective. *arXiv:2406.17969*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric. P Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. 2023. [Lmsys-chat-1m: A large-scale real-world llm conversation dataset](#). *Preprint*, arXiv:2309.11998.

A ReasonScore Details

A.1 Reasoning Vocabulary

In Fig. 5, we show the complete list of words from the *reasoning* vocabulary \mathcal{R} that we obtain in Sec. 3.1. For clarity, we list only the lowercase forms without spaces (e.g. “alternatively”). However, in our implementation, we track multiple forms of each word, including capitalized (“Alternatively”) and space-prefixed variants (“ alternatively”, “ Alternatively”), as the tokenizer can assign different tokens for each of the forms.

```
["alternatively", "hmm", "maybe", "wait",  
"perhaps", "let me", "therefore", "however",  
"but", "another"]
```

Figure 5: The complete list of words from the *reasoning* vocabulary \mathcal{R} in the lowercase and without spaces form.

A.2 ReasonScore Distribution

Fig. 6 shows the ReasonScore values sorted in decreasing order of all SAE features for DEEPSEEK-R1-LLAMA-8B. We select the 0.997-th quantile as a cutoff, yielding approximately 200 features. While the plot shows the ReasonScore continues to decay below this threshold rather than reaching a plateau, this amount is feasible to analyze manually and contains all the most important features as judged by our metric.

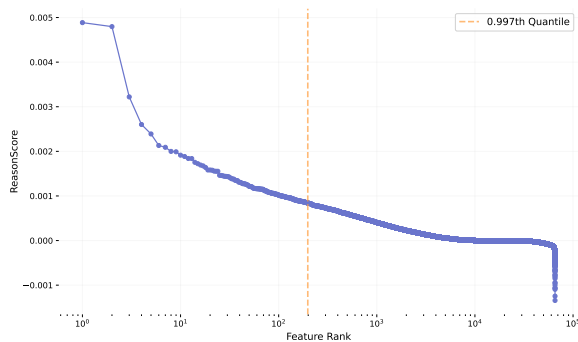


Figure 6: Distribution of ReasonScore values across all SAE features for DEEPSEEK-R1-LLAMA-8B, sorted in decreasing order.

B Interpretability Details

B.1 Feature Interfaces

Figs. 7,8 display additional activation patterns for features we found during manual interpretation

(Sec. 4.2), highlighting tokens where each feature most strongly activates.

B.2 Automatic Interpretation Details

We cluster reasoning-related features based on their possible functions and behavioral patterns. To support this process, we provided GPT-4o with a list of existing features, accompanied by a description of its possible function and observed steering behavior. The model was asked to identify recurring patterns and group similar features accordingly. To ensure accuracy, the results were then manually reviewed and validated. Table 2 presents the resulting feature groups, including categories (reasoning depth and thoroughness, self-correction and backtracking, and others), along with descriptions of their roles and effects. In some cases, even features grouped together based on shared function exhibited subtle differences in how they influenced responses; for instance, among features encouraging structural organization, one may focus on logical flow and paragraphing, while another influences transitions between argument steps. Additionally, features often demonstrated overlapping effects with other groups or influenced aspects beyond reasoning alone. For example, affecting the stylistic tone or structure of the output. This suggests that certain features may play a broader role across different types of reasoning and expression, rather than being confined to a single function.

C Examples of Feature Steering

Tab. 3 show the example of model’s thinking process on a “how many r’s in the word strawberry” question with and without steering. Tabs. 4,5,6 show the examples of model’s thinking processes on reasoning-related benchmarks with and without steering.

Feature #16778
a lower concentration? Wait, but the initial thought was possible of them. Wait, no. Wait, the smallest possible. \n\nWait, let's think of it number. Wait, no, let's compute that again

Feature #31052
a perfect fit. But I'm not confident. Alternatively 't fit better. But I'm really not sure.

the answer.\n\nBut I'm not entirely sure. Another conquer enemy, but I'm not certain. Alternatively,

Feature #43828
a_i+1.\n\nBut this is unclear.\n\nAlternatively +1=42. But this is speculative.\n\nAlternatively of that intersection point. But this seems too abstract.\n\n*4+3. But this is just a guess

Feature #45699
a bit of a dilemma. \n\nWait, let me This is a bit ambiguous. To avoid confusion, I perspective, disputes might be.\n\nBut without context, the So there's a conflict. To resolve this, perhaps

Feature #40262
of a ship, but maybe "keel" relates coffin" could be a pun on "c 't make sense. Or maybe a homophone. " 'm not sure. Alternatively, 500 in Roman numer

Feature #34967
think of possible groupings.\n\nAlternatively, maybe the groups is a guess.\n\nWait, maybe the user expects me a missing detail. Alternatively, maybe the pie chart was 'm not certain. Wait, the options are (A

Feature #1451
"exit."\n\nWait, another angle: the room's missed something. Let me think again. The standard electrode 't fit.\n\nWait, another thought: maybe the answer confusing. Wait, let me try another approach. The

Feature #33429
not the case. Alternatively, maybe the problem is expecting is known, so maybe the answer is simply -57 . Alternatively, maybe the problem is testing the knowledge that .\n\nWait, maybe the problem is a simple application of

(a) Top-activating examples from the manually verified set of features. The chosen examples represent “uncertainty”. (b) Top-activating examples from the manually verified set of features. The chosen examples represent “exploration”.

Figure 7: Examples of feature interfaces used in manual interpretation experiments.

Feature #506
barycentric coordinates... I think there's a enter can be given, but I might need to recall)\n\nBut this seems complicated. Alternatively, perhaps it's external angle bisectors.\n\nBut I'm not sure how

Feature #9636
P and Q.\n\nWait, but we have P(B be necessary for our problem. Let's return to the help with the inequality?\n\nWait, maybe use Cauch would take the triangle.\n\nBut the problem states that the

Feature #9977
of the rows.\n\nBut the problem states "any initial feasible. However, the problem states that N is presence of competitors. But the question specifically asks about temperature done after chlorination. But the question says

Feature #34370
seems straightforward. But let me verify with actual numbers to .\n\nWait, but let me double-check. The problem 1. But let me try another way to solve 3.\n\nWait, let me check my steps again to

Feature #34431
is 530 J.\n\nBut another approach: 5 10âg»Â².\n\nBut maybe the calculation allows for be 17. However, sometimes when the first digit 1680 J.\n\nHowever, in our precise calculation,

Feature #54382
121 is the smallest.\n\nBut wait, let's check a mistake in the reasoning.\n\nWait, but the problem as R=4. Hmm. What about R= 's a different answer.\n\nWait, let's consider another

Feature #56769
or Poisson's ratio. This is a problem because it's not mentioned here. Hmm. The user might knowing the refractive index? Wait, maybe there's natural frequency without knowing E. Maybe there is a standard

Feature #57334
not likely.\n\nWait, perhaps I'm making a mistake already present.\n\nWait, perhaps the reaction uses a different substitution should be meta. Unless there's some steric does that happen?\n\nAlternatively, maybe the first Cl goes

(a) Top-activating examples from the manually verified set of features. The chosen examples represent “reflection”. (b) Top-activating examples from the manually verified set of features. The chosen examples represent mixed behaviors.

Figure 8: Examples of feature interfaces used in manual interpretation experiments.

Group Name	Features	Possible Function	Effect Type	Observed Behavior
Reasoning Depth and Thoroughness	506, 4395, 9636, 23052, 30288, 33148, 54382, 61935	Controls multi-step analysis, iteration, and self-correction in problem-solving.	Stylistic & Structural, Semantic & Logical Consistency	Strengthening: Extensive step-by-step reasoning, multiple iterations, self-corrections. Weakening: Direct answers with minimal steps.
Numerical Accuracy and Validation	4990, 3466, 16778, 46379, 34813, 51765	Governs precision in calculations, unit conversions, and error-checking.	Semantic & Logical Consistency	Strengthening: Meticulous unit tracking, iterative re-evaluation. Weakening: Direct results with potential errors.
Exploration of Multiple Methods	22708, 62446	Encourages evaluating alternative approaches before finalizing solutions.	Semantic & Logical Exploration	Strengthening: Compares multiple strategies (e.g., DP vs. greedy). Weakening: Commits to the first viable method.
Structural and Logical Organization	57334, 43828, 45699, 49326, 17726, 46449, 41636, 40262	Ensures clarity, step-by-step breakdown, and logical flow. It may also balance executable code generation vs. verbal explanations.	Structural, Semantic & Instruction Clarity	Strengthening: Well-structured explanations. Weakening: Disorganized or fragmented reasoning.
Symbolic vs. Numerical Reasoning	48026, 34967, 34589	Balances theoretical/symbolic reasoning with direct numerical computation.	Semantic & Logical Consistency	Strengthening: Algebraic/theoretical frameworks. Weakening: Immediate numerical substitution.
Self-Correction and Backtracking	16778, 35337, 42609, 34431, 25953	Controls iterative refinement and error-checking.	Semantic & Logical Consistency	Strengthening: Multiple rounds of self-correction. Weakening: Commits to initial answers without revision.
Causal Chaining & Scientific Context	56769, 34370, 3261, 13457	Enforces clear multi-step causal linkages in science/environmental scenarios, modulates temporal reasoning, hypothetical alternatives and scenario simulation	Semantic (Causality)	Strengthening: yields explicit causal chains, regulates contrastive reasoning, gives clearer timeline-based reasoning. Weakening: results in loosely linked assertions or missing intermediate steps, omit historical or causal context
Edge Case and Constraint Handling	16343, 46691, 3942	Ensures validation of edge cases and constraints.	Semantic & Logical Consistency	Strengthening: Explicitly addresses edge cases. Weakening: Assumes valid inputs without verification.
Semantic Elaboration & Conceptual Depth	1451, 33429, 61104, 25485, 45981, 31052, 16441, 53560	Shapes depth of domain-specific explanations, analogies, trade-off discussions, and interdisciplinary links	Semantic (Elaboration & Breadth)	Strengthening: Adds conceptual depth through analogies, trade-offs, and multi-factor explanations. Weakening: Reduces to simple, surface-level or single-cause statements with minimal abstraction.
Other Behavior	48792, 9977, 20046, 12036, 32134	Include: influences engagement and conversational tone, assertiveness/redundancy/structure in text and terminology	Stylistic & Structural	Strengthening: Creates a more formal, robotic style with rigid structure and a high degree of confidence in statements. Weakening: Makes the style livelier and more conversational, with informal delivery, varied structure, and a moderate level of confidence that includes elements of uncertainty and flexibility.

Table 2: Reasoning clusters obtained using GPT-4O. Each cluster is defined by a particular type of reasoning (depth of analysis, numerical checking, code generation), the specific features involved, their hypothesized function in shaping the models’ output style and logic, and the observed behaviors that emerge when using the feature steering.

Problem: How many r's are in the word strawberry?

Generation (no steering): I need to determine how many times the letter "r" appears in the word "strawberry." First, I'll write down the word and break it into individual letters: S, T, R, A, W, B, E, R, R, Y. Next, I'll identify which of these letters are "r": the third, eighth, and ninth letters. Finally, I'll count these "r"s, which gives me a total of three.

Generation (steering): Okay, so I need to figure out how many r's are in the word "strawberry." Let me think about this step by step. First, I should probably write down the word and break it down into its individual letters. The word is spelled S-T-R-A-W-B-E-R-R-Y. Wait, let me make sure I have that right. Sometimes I get confused with the spelling. I think the correct spelling is "strawberry." So, let me write it out again: S, T, R, A, W, B, E, R, R, Y. Hmm, now that I write it, I see there are two R's after the E. Let me count the letters again to make sure I don't miss any. S (1), T (2), R (3), A (4), W (5), B (6), E (7), R (8), R (9), Y (10). So the word has 10 letters in total. Now, focusing on the R's. The third letter is R, that's one. Then after E, which is the seventh letter, there are two more R's: the eighth and ninth letters. So that's two more. Let me double-check the spelling again to make sure I'm not miscounting. Sometimes I might confuse the letters. I think it's S-T-R-A-W-B-E-R-R-Y, so yes, the two R's are towards the end. Wait, is there another R in the word? Let me think. The word is "strawberry." I think it's spelled with an R after the E, and another R after that. So that's two R's. I don't recall any other R's in the word. Let me go through each letter again: S, T, R, A, W, B, E, R, R, Y. So, yes, the R's are the third, eighth, and ninth letters. Wait, that's three R's. Did I miscount earlier? Wait, no, I think I made a mistake. Let me recount the letters with their positions. Let's write it out:

1. S
2. T
3. R
4. A
5. W
6. B
7. E
8. R
9. R
10. Y

So, the R's are at positions 3, 8, and 9. That makes three R's in total. I think I initially thought it was two, but upon a closer examination, it's actually three. Let me check another source or think about how the word is pronounced. The word "strawberry" is pronounced with a double R sound, so that would correspond to two R's, but when I break it down, I see that the letters are R, R. Wait, no, the letters are E, R, R. So, that's two R's after the E. So, the third letter is R, and then the eighth and ninth letters are R's as well. So that's three R's. Wait, I'm getting confused. Let me visualize the word again: S-T-R-A-W-B-E-R-R-Y. So, the letters are S, T, R, A, W, B, E, R, R, Y. So, the R appears at the third position, and then again at the eighth and ninth positions. That makes three R's. So, the correct count is three R's. I think I initially miscounted because I focused on the double R at the end, but there's another R earlier in the word. So, to clarify, the word "strawberry" has three R's: one in the third position, and two more at the end. So, the correct answer is three R's.

Table 3: An example of "How many r's are in the word strawberry?" problem and corresponding full outputs from DEEPSEEK-R1-LLAMA-8B and its steered version.

Problem: For how many values of x is the expression $\frac{x-5}{x^2-9}$ not defined?

Generation (no steering): Okay, so I have this problem here: "For how many values of x is the expression $\frac{x-5}{x^2-9}$ not defined?" Let me try to figure this out step by step. First, I remember that a fraction is undefined when its denominator is zero because you can't divide by zero. So, to find where this expression is not defined, I need to figure out when the denominator equals zero. The denominator here is $x^2 - 9$. So, I need to solve the equation $x^2 - 9 = 0$. Hmm, how do I solve that? Well, I think it's a difference of squares because x^2 is like $(x)^2$ and 9 is $(3)^2$. So, the equation factors into $(x-3)(x+3) = 0$. That makes sense because when you multiply those two binomials, you get $x^2 - 3x + 3x - 9 = x^2 - 9$, which is correct. So, setting each factor equal to zero gives the solutions:

1. $x - 3 = 0$ leads to $x = 3$
2. $x + 3 = 0$ leads to $x = -3$

Therefore, the denominator is zero when $x = 3$ or $x = -3$. That means the expression $\frac{x-5}{x^2-9}$ is undefined at these two points. Wait, let me double-check. If I plug $x = 3$ into the denominator: $3^2 - 9 = 9 - 9 = 0$. Yep, that's zero. Same with $x = -3$: $(-3)^2 - 9 = 9 - 9 = 0$. So, both of these make the denominator zero. Is there a chance I missed any other values? Hmm, let me think. The denominator is a quadratic equation, which can have at most two real roots. Since we found both roots, 3 and -3, there are only two values where the denominator is zero. What about the numerator? The numerator is $x - 5$. That's a linear term, so it only affects the value of the expression, not its definition. So, the numerator doesn't cause any undefined behavior on its own. It's only when the denominator is zero that the whole expression becomes undefined. Therefore, the only values of x where the expression is not defined are $x = 3$ and $x = -3$. That means there are two values of x for which the expression is undefined. Just to make sure, let me consider the original expression again: $\frac{x-5}{x^2-9}$. So, the denominator is $x^2 - 9$, which factors to $(x-3)(x+3)$. So, if I were to graph this function, it would have vertical asymptotes at $x = 3$ and $x = -3$, which are the points where the function is undefined. There's no division by zero anywhere else, so these are the only two points where the function doesn't exist. I don't think I made any mistakes here. The problem is pretty straightforward once you recognize that the denominator needs to be factored and solved. So, yeah, I feel confident that the answer is two values: $x = 3$ and $x = -3$.

****Final Answer****

The expression is not defined for 2 values of x .

Generation (steering): Okay, so I have this problem: I need to find how many values of x make the expression $(x - 5)/(x^2 - 9)$ undefined. Hmm, let me think. I remember that a fraction is undefined when its denominator is zero because you can't divide by zero. So, I should figure out when $x^2 - 9$ equals zero. Alright, let me write that down. The denominator is $x^2 - 9$. So, I need to solve $x^2 - 9 = 0$. Hmm, that's a quadratic equation. I think I can factor this. Let me recall, the difference of squares formula is $a^2 - b^2 = (a - b)(a + b)$. So, applying that here, $x^2 - 9$ is the same as $x^2 - 3^2$. Therefore, it factors into $(x - 3)(x + 3)$. So, now I have $(x - 3)(x + 3) = 0$. To find the values of x that make this equation true, I can set each factor equal to zero. That gives me:

$$x - 3 = 0 \text{ or } x + 3 = 0$$

Solving these, I get $x = 3$ and $x = -3$. So, these are the values that make the denominator zero, which would make the entire expression undefined. Therefore, I need to check if these are the only values or if there are more. Wait, let me make sure. The original expression is a rational expression, so the only way it's undefined is if the denominator is zero. There are no other denominators or functions here, so I think that's all. Therefore, x cannot be 3 or -3. Let me just double-check my factoring. $x^2 - 9$: if I expand $(x - 3)(x + 3)$, I get $x*x + x*3 - 3*x - 3*3$, which is $x^2 + 3x - 3x - 9$, which simplifies to $x^2 - 9$. Yep, that's correct. So, are there any other possibilities where the denominator could be zero? Well, if I consider the quadratic equation $x^2 - 9 = 0$, it only has two roots: 3 and -3. There's no other x -value that can satisfy this equation because a quadratic equation can have at most two real roots. So, that's it. There are only two values of x that make the denominator zero, hence the expression is undefined for those x . Wait, let me think again. Maybe I should plug these x -values into the original expression to confirm. Let's try $x = 3$ first. Plugging in, the denominator becomes $3^2 - 9 = 9 - 9 = 0$. So, the expression becomes $(3 - 5)/0$, which is $(-2)/0$. That's undefined, so that's correct. Now, $x = -3$. Plugging in, the denominator is $(-3)^2 - 9 = 9 - 9 = 0$ again. So, the expression becomes $(-3 - 5)/0$, which is $(-8)/0$. That's also undefined. So, both values definitely make the expression undefined. Is there any other way this could be undefined? For example, sometimes, if the numerator is also zero, I might have an indeterminate form, but in this case, let's see. When $x = 3$, the numerator is $3 - 5 = -2$, which is not zero. When $x = -3$, the numerator is $-3 - 5 = -8$, which is also not zero. So, both times, the numerator is a non-zero number, and the denominator is zero. Therefore, it's just undefined, not something like $0/0$, which is actually undefined as well, but in that case, both numerator and denominator are zero. But in this case, it's just a clear division by zero. Therefore, I can confidently say that the expression is undefined for $x = 3$ and $x = -3$. So, how many values is that? Let me count. 3 and -3. That's two different values. So, the number of values where the expression is undefined is 2. Wait, let me just visualize the graph of the function. If I were to graph $y = (x - 5)/(x^2 - 9)$, I know that the function will have vertical asymptotes where the denominator is zero, provided that the numerator isn't zero there. So, as I found earlier, at $x = 3$ and $x = -3$, there are vertical asymptotes because the denominator is zero, and the numerator isn't zero. So, those points are not in the domain of the function. So, that's another way to confirm. Is there a possibility of any other points where the function is undefined? Let me think. Since the denominator is a quadratic, it can have at most two real roots, and we've already found both. So, nope, no other points where the denominator is zero. So, that should be it. So, summarizing, the expression is undefined when x is 3 or -3. Therefore, there are two values of x where the expression is undefined. So, the answer should be 2. Wait, just to make sure I didn't make any mistake in my calculations. Let me go through the steps again.

1. Start with the expression $(x - 5)/(x^2 - 9)$.
2. To find when it's undefined, set denominator equal to zero: $x^2 - 9 = 0$.
3. Factor the quadratic: $(x - 3)(x + 3) = 0$.
4. Solve for x : $x = 3$ and $x = -3$.
5. Check if these x -values make the numerator zero as well. For $x = 3$: numerator is $3 - 5 = -2 \neq 0$. For $x = -3$: numerator is $-3 - 5 = -8 \neq 0$. So, both are valid points where the expression is undefined.
6. Confirm that these are the only points by noting that a quadratic equation can have at most two real roots, and we found both.
7. Visualize the graph to check for vertical asymptotes at these points, which confirms the function is undefined there.

Therefore, I'm pretty confident that the number of values is 2.

****Final Answer****

The expression is undefined for 2 values of x .

Table 4: A problem from MATH-500, and corresponding outputs from DEEPSEEK-R1-LLAMA-8B and its steered version.

Problem: A regular octagon has the same perimeter as the regular hexagon shown here with side length 16 cm. How long is each side of the octagon? [asy]size(80); pair A = dir(120); pair B=dir(60); pair M=(A+B)/2; draw(dir(360)-B-A-dir(180)-dir(240)-dir(300)-cycle); label("16 cm", M, N);[/asy]

Generation (no steering): Okay, so I have this problem where a regular octagon has the same perimeter as a regular hexagon. The hexagon has a side length of 16 cm, and I need to find the side length of the octagon. Hmm, let's break this down step by step. First, I remember that the perimeter of a regular polygon is just the number of sides multiplied by the length of each side. So, for the hexagon, which is a six-sided polygon, I can calculate its perimeter by multiplying 6 by 16 cm. Let me write that out:
Perimeter of hexagon = $6 \times 16 \text{ cm} = 96 \text{ cm}$.

Got that. So the perimeter of the octagon must also be 96 cm since it's equal to the hexagon's perimeter. Now, a regular octagon has 8 sides, right? So, if I let 's' be the length of each side of the octagon, then the perimeter of the octagon would be $8 \times s$. Since we know the perimeter is 96 cm, I can set up the equation:
 $8 \times s = 96 \text{ cm}$.

To find 's', I just need to divide both sides by 8:

$s = 96 \text{ cm} / 8 = 12 \text{ cm}$.

Wait, is that it? It seems straightforward, but let me double-check to make sure I didn't miss anything. So, the hexagon has 6 sides, each 16 cm, so $6 \times 16 = 96 \text{ cm}$. The octagon has 8 sides, so each side is $96/8 = 12 \text{ cm}$. Yep, that seems correct. Hmm, just to make sure, maybe I should visualize the polygons. A regular hexagon has six equal sides and six equal angles, and a regular octagon has eight equal sides and eight equal angles. Since both are regular, all sides and angles are equal, so the calculation should hold.

Another way to think about it: the perimeter is just the sum of all sides. So, for the hexagon, adding up six sides each of 16 cm gives 96 cm. For the octagon, we need to add up eight sides, each of which is the same length, so 8 times that length equals 96 cm. Solving for that length gives 12 cm. I don't see any mistakes in the logic here. It's a direct calculation based on the definition of perimeter for regular polygons. So, I think the side length of the octagon is 12 cm.

****Final Answer****

The length of each side of the octagon is 12 centimeters.

Generation (steering): Okay, so I have this problem here where I need to figure out the length of each side of a regular octagon. The octagon has the same perimeter as a regular hexagon, and the hexagon is shown with side length 16 cm. Hmm, okay, let me think step by step about how to approach this. First, I remember that the perimeter of a shape is the total length around it. For regular polygons, which both octagons and hexagons are, the perimeter is just the number of sides multiplied by the length of each side. So if I can figure out how many sides each shape has and then use the given information, I can find the opposite. Wait, the problem says it's a regular hexagon. Right, a hexagon has six sides. So, since the side length is 16 cm, the perimeter of the hexagon would be 6 times 16 cm. Let me write that down.

Perimeter of hexagon = $6 \times 16 \text{ cm} = 96 \text{ cm}$.

So, the octagon has the same perimeter, which is 96 cm. Now, an octagon has eight sides. So, if I can find the length of each side of the octagon, that would be the perimeter divided by 8. Let me write that formula:

Perimeter of octagon = $8 \times (\text{length of each side})$

But wait, we know the perimeter is 96 cm, so:

$96 \text{ cm} = 8 \times (\text{length of each side})$

To find the length of each side, I need to divide both sides by 8. So, length of each side = $96 \text{ cm} / 8 = 12 \text{ cm}$. Hmm, that seems straightforward. But let me just double-check to make sure I didn't make a mistake.

Wait, is it correct that the octagon's perimeter is 96 cm? Because the hexagon's side is 16 cm, so 6 times 16 is 96. So, yes, the octagon has to have the same perimeter. Since octagons have more sides, each side should be shorter. So, 96 divided by 8 is 12, which is shorter than 16, so that makes sense. Wait, another way to think about it: if you have a polygon with more sides, each side is actually shorter for the same perimeter. So, since an octagon has more sides than a hexagon, each side should be shorter, which aligns with 12 cm being shorter than 16 cm. So that seems right. Is there another way to approach this problem?

Maybe using the formula for perimeter in terms of side length? Yes, for regular polygons, the perimeter is just the number of sides multiplied by the side length. So, if I denote the side length of the octagon as 's', then:

Perimeter of octagon = $8 \times s$

But since it's equal to the perimeter of the hexagon, which is $6 \times 16 = 96$, then: $8s = 96$ Solving for s gives $s = 96 / 8 = 12 \text{ cm}$. Yep, same result. I think I got it. So, the length of each side of the octagon is 12 cm. Wait, let me think again. Sometimes, when I'm dealing with perimeters, I might confuse side length with something else, but in this case, since both are regular polygons, it's straightforward. So, for regular polygons, the perimeter is directly additive. So, each side contributes equally. Therefore, inverting the number of sides inversely affects the side length. So, in the hexagon, 6 sides, each 16 cm, so 16 times 6 is 96. The octagon, which has more sides, 8, so each side is 96 divided by 8, which is 12. So, that makes sense. Wait, just to make sure, if the octagon had 8 sides, each 12 cm, the perimeter would be $8 \times 12 = 96 \text{ cm}$, which is the same as the hexagon. So, that's correct. I think that's solid. So, the length of each side of the octagon is 12 cm.

****Final Answer****

The length of each side of the octagon is 12 cm.

Table 5: A problem from MATH-500, and corresponding outputs from DEEPSEEK-R1-LLAMA-8B and its steered version.

Problem: Thirty-five students attend a math club meeting. The number of girls at the meeting is a multiple of 13, and there are more girls than boys attending the meeting. How many boys are at the meeting?

Generation (no steering): Alright, so I have this problem here: There are 35 students at a math club meeting. The number of girls is a multiple of 13, and there are more girls than boys. I need to find out how many boys are at the meeting. Hmm, okay. Let's break this down. First, total students are 35. Girls are a multiple of 13, so let's denote the number of girls as G . Then, the number of boys would be B . So, $G + B = 35$. Also, it's given that there are more girls than boys, so $G > B$. And since G is a multiple of 13, let's think about what multiples of 13 are possible here. 13 times 1 is 13, 13 times 2 is 26, 13 times 3 is 39. Wait, but 39 is more than 35, so that can't be the number of girls because we only have 35 students in total. So, possible number of girls are 13 or 26. So, G could be 13 or 26. Let's check both possibilities. Case 1: $G = 13$. Then, $B = 35 - 13 = 22$. So, number of boys would be 22. But wait, the problem says there are more girls than boys. So, 13 girls and 22 boys. That would mean more boys than girls, right? Because 22 is more than 13. That contradicts the given condition. So, this case is not possible. Case 2: $G = 26$. Then, $B = 35 - 26 = 9$. So, number of boys would be 9. Now, 26 girls and 9 boys. That means more girls than boys, which fits the condition. So, this is the valid case. Wait, let me double-check. If $G = 26$, which is 2 times 13, then B is 9. $26 + 9 = 35$, correct. And $26 > 9$, so the condition holds. So, that seems to be the answer. Is there any other multiple of 13 that I need to consider? Let's see. The next multiple is 39, but as I thought earlier, that's more than 35, so that's not possible. So, the only possible number of girls is 26, leading to 9 boys. Alternatively, let's approach this algebraically. Let me write the equations:

$$G + B = 35$$

G is a multiple of 13, so $G = 13k$, where k is an integer. Also, $G > B$, so $13k > B$. But since $G + B = 35$, then $B = 35 - G = 35 - 13k$. So, substituting back into $G > B$:

$$13k > 35 - 13k$$

So, let's solve this inequality:

$$13k > 35 - 13k$$

Adding $13k$ to both sides:

$$26k > 35$$

Divide both sides by 26:

$$k > 35/26$$

35 divided by 26 is approximately 1.346.

Since k must be an integer, the smallest possible k is 2. So, $k = 2$, which gives $G = 26$, then $B = 35 - 26 = 9$. So, that's the same result as before. Alternatively, if k was 1, which gives $G = 13$, then $B = 22$, but that doesn't satisfy $G > B$, so k must be 2. Hence, the number of boys is 9. I think that's thorough. I considered both cases, checked the algebra, and made sure that the conditions are satisfied. So, the answer should be 9 boys.

****Final Answer****

The number of boys at the meeting is 9.

Generation (steering): Okay, so I have this problem here: there are 35 students at a math club meeting. The number of girls is a multiple of 13, and there are more girls than boys. I need to find out how many boys are at the meeting. Hmm, let me try to break this down. First, let me note down the given information:

- Total number of students = 35.

- Number of girls is a multiple of 13.

- Number of girls is more than the number of boys.

I need to find the number of boys. Let me denote the number of girls as G and the number of boys as B . So, $G + B = 35$.

Since G is a multiple of 13, that means $G = 13 * k$, where k is a positive integer. Because there are more girls than boys, $G > B$. So, since $G + B = 35$, if G is more than B , then G must be greater than $35/2$, which is 17.5. So, G must be at least 18. But since G is a multiple of 13, let's see which multiples of 13 are greater than 17.5.

Multiples of 13: 13, 26, 39, etc. But since the total number of students is 35, G can't be 39 because that would mean more students than the total. So, the possible multiples of 13 for G are 13 and 26. But wait, earlier I thought G must be more than 17.5, so 13 is less than that, which would mean G is 13. But 13 is less than 17.5, so that would mean B is 22, which is more than G . But the problem says more girls than boys, so G must be greater than B . So, G can't be 13 because that would make $B = 35 - 13 = 22$, which is more. So, actually, G must be 26. Wait, let me think again. If G is 26, then B would be $35 - 26 = 9$. So, $G = 26$, $B = 9$. But is 26 a multiple of 13? Yes, 26 is $13 * 2$. So that seems to fit. Also, 26 is more than 9, which satisfies the condition that there are more girls than boys. But hold on, let me double-check. If G is 26, then B is 9, so $26 + 9 = 35$. Yes, that adds up. So, is 26 the only possible multiple of 13? Let me check the next multiple. $13 * 3$ is 39, which is more than 35, so that can't be. So, the only possible multiple of 13 is 26 for G . Therefore, the number of boys is $35 - 26 = 9$. Wait, but hold on, is 9 the number of boys? Let me confirm. If G is 26, which is a multiple of 13, and $26 > 9$, so yes, that makes sense. So, the number of boys is 9. Wait, is there another way to think about this? Let me approach it algebraically. Let me write $G = 13k$, where k is a positive integer. So, since $G + B = 35$, then $B = 35 - G = 35 - 13k$. We know that $G > B$, so $13k > 35 - 13k$. Let's solve for k :

$$13k > 35 - 13k$$

Adding $13k$ to both sides:

$$26k > 35$$

Divide both sides by 26:

$$k > 35 / 26$$

Calculating $35 / 26$, which is approximately 1.346. Since k must be an integer, the smallest possible value of k is 2. So, $k = 2$:

$$G = 13 * 2 = 26$$

$$\text{Then, } B = 35 - 26 = 9.$$

If k were 3, G would be 39, which is more than the total number of students, which is 35. So, that's not possible. Therefore, the only possible value is $k=2$, leading to $G=26$ and $B=9$. Wait, so is 9 the number of boys? That seems consistent. Let me just double-check my reasoning. If there are 26 girls, that's a multiple of 13, more than the number of boys, which is 9. $26 + 9$ is 35, so that adds up. There are no other multiples of 13 in the opposite direction because the next multiple would be 39, which is too high. So, I think I've got it. The number of boys is 9. But wait, just to make sure, let me think about it again. If G is 13, then B is 22. But since G has to be more than B , that's not possible because $13 < 22$. So, G cannot be 13 because that would make more boys than girls, which contradicts the given condition. So, G must be 26, which makes B 9. Therefore, the number of boys is 9.

****Final Answer****

The number of boys at the meeting is 9.

Table 6: A problem from MATH-500, and corresponding outputs from DEEPSEEK-R1-LLAMA-8B and its steered version.

Problem: A curve is parameterized by

$$(x, y) = (t^3 + 7, -3t^2 - 6t - 5).$$

Find the point the curve passes through at $t = 2$.

Generation (no steering): Okay, so I have this problem here where a curve is parameterized by the equations $x = t^3 + 7$ and $y = -3t^2 - 6t - 5$. I need to find the point the curve passes through when $t = 2$. Hmm, let me think about how to approach this. First, I remember that when you have a parameterized curve, each value of t corresponds to a specific point (x, y) on the curve. So, to find the point when $t = 2$, I just need to substitute $t = 2$ into both the equations for x and y and compute the values. That makes sense because each parameter t defines a unique point on the curve. Alright, let's start with the equation for x . The equation is $x = t^3 + 7$. So, substituting $t = 2$ into this should give me the x-coordinate of the point. Let me compute that step by step. Calculating t^3 when $t = 2$: $2^3 = 8$. Then, adding 7 to that: $8 + 7 = 15$. So, the x-coordinate is 15. That seems straightforward. Now, moving on to the y-coordinate. The equation is $y = -3t^2 - 6t - 5$. Again, substituting $t = 2$ into this equation. Let me break it down. First, compute t^2 when $t = 2$: $2^2 = 4$. Then, multiply that by -3 : $-3 \times 4 = -12$. Next, compute $-6t$ when $t = 2$: $-6 \times 2 = -12$. Lastly, we have the constant term, which is -5 . So, putting it all together: $y = (-12) + (-12) + (-5)$. Wait, hold on. Let me double-check that. The equation is $y = -3t^2 - 6t - 5$. So, substituting $t = 2$, it's $-3(2)^2 - 6(2) - 5$. That is $-3 \times 4 - 12 - 5$. So, $-12 - 12 - 5$. Adding those together: $-12 - 12$ is -24 , and $-24 - 5$ is -29 . So, the y-coordinate is -29 . Let me make sure I did that correctly. So, $-3(2)^2$ is $-3 \times 4 = -12$. Then, -6×2 is -12 , and then subtracting 5. So, adding all those up: $-12 - 12$ is -24 , minus another 5 is -29 . Yeah, that seems right. So, combining both coordinates, when $t = 2$, the point on the curve is $(15, -29)$. Hmm, let me just verify once more in case I made any calculation errors. Starting with x : $t^3 + 7$. $t = 2$, so 2^3 is 8, plus 7 is 15. That's correct. For y : $-3t^2 - 6t - 5$. $t = 2$: $-3(4) = -12$, $-6 \times 2 = -12$, and the constant is -5 . So, adding them: $-12 - 12$ is -24 , minus 5 is -29 . That's correct. Okay, so I think that's the point. Just to visualize, when $t = 2$, the curve is at $(15, -29)$. I don't see any mistakes in my calculations, so I feel confident about this answer.

****Final Answer****

The point the curve passes through at $t = 2$ is $(15, -29)$.

Generation (steering): Alright, so I have this problem here where a curve is parameterized by the equations $x = t^3 + 7$ and $y = -3t^2 - 6t - 5$. I need to find the point the curve passes through when $t = 2$. Hmm, okay, let me think about how to approach this. First, I remember that parameterized equations give both the x and y coordinates in terms of a third variable, which in this case is t . So, when they ask for the point when $t = 2$, I need to substitute $t = 2$ into both equations for x and y and then find the corresponding (x, y) pair. Let me write that down step by step. So, for x , the equation is $x = t^3 + 7$. If I plug in $t = 2$, that should give me the x-coordinate. Similarly, for y , the equation is $y = -3t^2 - 6t - 5$, so substituting $t = 2$ there should give me the y-coordinate. Once I have both values, I can combine them into a point. Okay, let me start with calculating x . Plugging $t = 2$ into $x = t^3 + 7$:

$$x = (2)^3 + 7.$$

Calculating 2^3 , that's 8, right? So, $8 + 7$ is 15. So, the x-coordinate when $t = 2$ is 15. That seems straightforward. Now, moving on to the y-coordinate. The equation is $y = -3t^2 - 6t - 5$. Plugging in $t = 2$: $y = -3(2)^2 - 6(2) - 5$.

Let me compute each term step by step. First, 2^2 is 4, so multiplying that by -3 gives -12 . Next, 6×2 is 12, so with the negative sign, that term becomes -12 . Then, the last term is just -5 . So, putting it all together: $-12 - 12 - 5$. Let's add those numbers. $-12 - 12$ is -24 , and then subtracting another 5 gives -29 . So, the y-coordinate is -29 . Wait, let me double-check that calculation because sometimes when I have multiple negative signs, I might mix them up. So, let me recast it:

$$y = -3(4) - 6(2) - 5.$$

That's $-12 - 12 - 5$. Yes, that's correct. $-12 - 12$ is indeed -24 , and $-24 - 5$ is -29 . So, that seems right. So, to recap, when $t = 2$, the x-coordinate is 15 and the y-coordinate is -29 . Therefore, the point the curve passes through is $(15, -29)$. Wait, just to make sure I didn't make a mistake in the substitution, let me verify each step again. For x , substituting $t = 2$ into $t^3 + 7$: $2^3 = 8$, so $8 + 7 = 15$. That's correct. For y , substituting $t = 2$ into $-3t^2 - 6t - 5$:

$$\text{First term: } -3(2)^2 = -3 \times 4 = -12.$$

$$\text{Second term: } -6 \times 2 = -12.$$

$$\text{Third term: } -5.$$

Adding them up: $-12 - 12 - 5 = (-12 - 12) - 5 = -24 - 5 = -29$. Yep, that's correct. Hmm, I think that's solid. Maybe I should just think about whether the parameterization makes sense or if there's another way to represent this curve. Wait, actually, I might recall that sometimes parameterizations can be tricky, but in this case, it's pretty straightforward. It's giving x and y in terms of t , so plugging in $t = 2$ should give the point. Alternatively, I can also compute the values for $t = 2$ in both equations and plot them mentally or on paper. Let me sketch a rough graph in my mind. So, when $t = 2$, x is 15, which is further out on the x-axis, and y is -29 , which is quite low on the y-axis. That seems plausible for a cubic and quadratic parameterization. Wait, another thought: sometimes, when I parameterize, especially with higher powers, I might have to check for consistency, but in this case, since both x and y are given, and the question is just about plugging in a specific t , I think it's safe. So, to sum up, substituting $t = 2$ into both equations gives $x = 15$ and $y = -29$, so the point is $(15, -29)$. I think that's the correct answer.

****Final Answer****

The point the curve passes through at $t = 2$ is $(15, -29)$.

Table 7: A problem from MATH-500, and corresponding outputs from DEEPSEEK-R1-LLAMA-8B and its steered version.