ReflCtrl: Controlling LLM Reflection via Representation Engineering

Anonymous Author(s)
Affiliation
Address
email

Abstract

Large language models (LLMs) with Chain-of-Thought (CoT) reasoning have achieved strong performance across diverse tasks, including mathematics, coding, and general reasoning. A distinctive ability of these reasoning models is **self-reflection**: the ability to review and revise previous reasoning steps. While self-reflection enhances the reasoning performance, it also increases inference cost. In this work, we study self-reflection through the lens of **representation engineering**. We segment model's reasoning into steps, identify those corresponding to reflection, and extract a reflection direction in the latent space that governs this behavior. Using this direction, we propose a stepwise steering method that can control reflection frequency. We call our framework ReflCtrl. Our experiments show that (1) for many cases the reflections are redundant, especially in stronger models. In our experiment, we can save up to 33.6% while preserving the performance. (2) model's reflection behavior is highly correlated with internal uncertainty signal, implying self-reflection may be controlled by model's uncertainty.

1 Introduction

Large language models (LLMs) have shown great success in many reasoning-related tasks, including math, coding, and general reasoning. A common technique for enhancing LLM reasoning is Chain-of-Thought (CoT) prompting [Wei et al., 2022], which asks the model to decompose the reasoning process into intermediate steps. Recently, a new class of models has been trained to develop **native reasoning ability**, such as OpenAI o1 [OpenAI, 2024] and Deepseek-r1 [DeepSeekAI et al., 2025]. They can automatically generate reasoning steps before providing a response, even without being prompted to do so.

Notably, these reasoning models develop the ability to **self-reflect**, i.e. rethink their previous reasoning during training. This is described by Deepseek-R1 [DeepSeekAI et al., 2025] as the "aha moment". The self-reflection ability is a key difference between reasoning models and their non-reasoning counterparts, and is widely believed to be a reason for boosted reasoning ability. Additionally, it is also a costly component in inference: our empirical study finds self-reflection consumes 25-30% of total reasoning tokens.

Despite its potential importance, the mechanism underlying self-reflection has not been well understood. In this work, we investigate this phenomenon through the lens of representation engineering [Zou et al., 2023], focusing on two central research questions:

RQ1: When does the model initiate reflection during its reasoning process?

RQ2: How does reflection influence the model's reasoning performance?

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

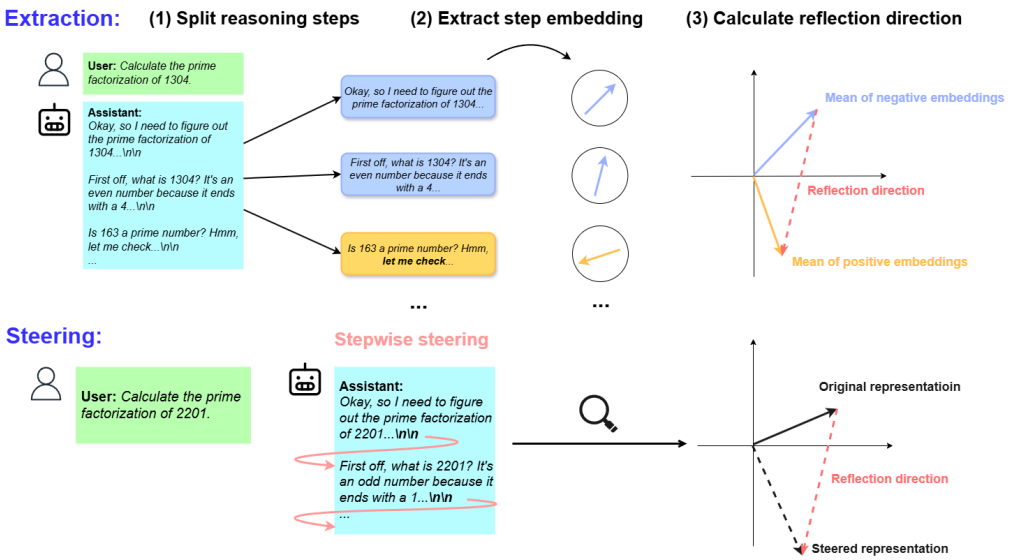


Figure 1: **Overview of the proposed ReflCtrl framework.** The model's reasoning is first segmented into steps, then reflection-related steps are identified through keywords. Finally, a reflection direction is extracted by calculating mean difference in the latent space. This direction can be used to steer the model's self-reflection behavior via the proposed stepwise steering method, enabling control of reflection frequency and inference cost.

To answer these questions, we propose a novel method to identify the **reflection direction** in the model's latent representation space. Steering experiments demonstrate that this direction can effectively control the number of reflections during reasoning. Our empirical analysis further reveals that in many cases, the model's reflections are redundant, offering an opportunity to reduce computational cost without sacrificing accuracy. Our contributions can be summarized as:

1. We identify reflection direction in the model's representation space that controls self-reflection, enabling steering model's reflection behavior according to user's intention.
2. We connect the model's reflection direction to model's internal uncertainty. Using model's representation projection on reflection direction as features, we show the performance is better than using representation from last layer. This implies model's reflection behavior may be controlled by internal uncertainty measurement.
3. Utilizing reflection direction we discover, we steer model's reflection to analyze the impact of reflections on reasoning performance. Empirical results suggests in many cases model's self-reflection could be redundant. Further, we design a novel stepwise steering method to address reflection redundancy. This new method reduces inference cost while preserving reasoning performance.

2 Related works

Reasoning LLMs Motivated by the success of Chain-of-Thought reasoning, several models have been trained to enhance native reasoning capability by generating thinking steps. OpenAI's o1 [OpenAI, 2024] leverages reinforcement learning to deliberate thinking during inference. DeepSeekAI et al. [2025] introduces a more cost-efficient training method with the Grouped Relative Policy Optimization (GRPO) algorithm, as well as its distilled variants (Deepseek-r1-distilled) that equip smaller models with thinking ability. QwQ-32b [Team, 2024] is a medium-sized reasoning model that achieves competitive performance with Deepseek-r1 and o1. In this work, we focus on QwQ-32b [Team, 2024] and Deepseek-r1-distill series, as they are open-sourced, allowing us to apply representation engineering techniques.

Representation engineering on LLMs While modern LLMs demonstrate remarkable capabilities, their internal mechanism is still not fully understood. Representation engineering [Zou et al., 2023, Bartoszcze et al., 2025] provides a tool to understand and steer model behavior by manipulating internal representations. Zou et al. [2023] shows that representation engineering can be applied to multiple safety-related aspects by reading and editing model's internal representation. With the rise of reasoning models, representation engineering methods specialized for these models have emerged: ThinkEdit [Sun et al., 2025] identifies a set of neurons controlling "short-thinking" and mitigates it via weight editing. Wang et al. [2025b] identifies special experts that coordinate reasoning and improves models' reasoning performance with a training-free method called RICE. In contrast, our work focuses on the **reflection behavior** of reasoning LLMs, which is an interesting reasoning pattern introduced in reinforcement learning but not yet systematically investigated from a representation engineering perspective.

Self-reflection. DeepSeekAI et al. [2025] report that models learn to self-reflect autonomously, described as the "aha moment". Yang et al. [2025] examines this phenomenon by comparing reasoning models with their non-reasoning counterparts in terms of linguistic patterns and description of uncertainty. Wang et al. [2025a] proposed reducing excessive reflection by suppressing corresponding tokens to reduce models' overthinking. In this work, we adopt a representation engineering perspective, revealing that model's reflection is correlated with its internal uncertainty representation and can be directly controlled via our proposed method.

3 Probing and steering self-reflection

In this section, we investigate reflection behavior in reasoning models through the lens of representation engineering. We start by identifying reflection steps in the model's reasoning, then extract a reflection direction in the latent space, and finally use this direction to steer model's behavior.

3.1 Background

Reasoning LLMs are built upon the Transformer decoder architecture [Vaswani et al., 2017], which stacks multiple identical layers. Each decoder layer l processes the hidden representation $z_l \in \mathbb{R}^d$. It consists of two major components: a self-attention block and a feed-forward MLP block. Formally, it can be written as:

$$\begin{aligned} \tilde{z}_l &= z_l + z_l^{\text{attn}},\ z_l^{\text{attn}} = \text{Attn}(\text{LN}(z_l)), \\ z + l + 1 &= \tilde{z}_l + z_l^{\text{mlp}},\ z_l^{\text{mlp}} = \text{MLP}(\text{LN}(z_l)). \end{aligned} \tag{1}$$

Here, $\text{LN}(\cdot)$ denotes layer normalization, $\text{Attn}(\cdot)$ is the self-attention block and $\text{MLP}(\cdot)$ is the feed-forwards network. We denote $\tilde{z}_l$ as the intermediate state after the attention block.

3.2 Identify reflection behavior

Reasoning LLMs usually produce a long, multi-step thinking process. To facilitate our analysis of model's reasoning, we first split the generated reasoning into thinking steps. We observe that such steps are naturally separated by the token sequence "\n\n" (an empty line) in most reasoning models, with each segment representing a coherent chunk of reasoning. Therefore, we treat each segment separated by "\n\n" as the smallest unit of analysis.

To identify reflection steps, we search for specific keywords within each step that mark the start of a new reflection, e.g., "Let me think", "Wait". While a reflection may span multiple steps, we identify it by detecting their initial step containing these keywords.

3.3 Extract reflection direction

With labeled reflection steps, we next compute the reflection direction in the latent space. For each step s at layer l, we extract all internal representations from the MLP and attention output of the first token, denoted as $z_l^{\{\text{attn,mlp}\}}(s)$. We use the first token because it captures the model's internal state when reflection is initialized, allowing us to investigate the triggering mechanism of reflections. The reflection direction is then defined as the mean difference between reflection and non-reflection

embeddings:

$$d_l^{\{\text{attn,mlp}\}} = \frac{1}{|\text{R}|} \sum_{s \in \text{R}} z_l^{\{\text{attn,mlp}\}}(s) - \frac{1}{|\text{NR}|} \sum_{s \in \text{NR}} z_l^{\{\text{attn,mlp}\}}(s), \tag{2}$$

where R and NR are the sets of reflection and non-reflection steps, respectively.

3.4 Steer model's reasoning

With the reflection direction, we can steer the model's reasoning by injecting this direction into its internal representations. Specifically, the intervention is applied by directly adding the direction vector:

$$z_{l,\text{intv}}^{\{\text{attn,mlp}\}} = z_l^{\{\text{attn,mlp}\}} + \lambda d_l^{\{\text{attn,mlp}\}}. \tag{3}$$

Here, z_l denotes the model output at layer l.

In standard representation-steering approaches, the intervention is applied at **every token generation step**. However, at high intervention strengths, this may push the model's representation far from the training distribution, thereby hurting performance.

To address this, we propose **stepwise steering**: instead of applying intervention on every token, we apply it only when the model **begins a new thinking step**. Specifically, the intervention is triggered when the last generated token matches the step delimiter "\n\n". As shown in figs. 4a and 4b, this method preserves intervention effects while avoiding the performance drop observed in full-token steering at high intervention strengths, allowing users to have more control on inference tokens without sacrificing performance.

3.5 Probing model's uncertainty on reflection direction

Another application of reflection direction is to investigate **RQ1** we proposed in Sec. 1: *When will self-reflection be triggered?* Our hypothesis is:

Reasoning LLMs trigger reflection when its internal uncertainty is high.

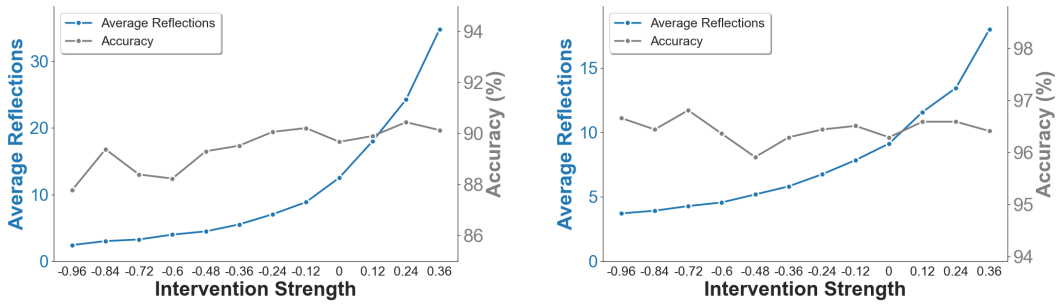
To verify, we need an approach to quantify the model's uncertainty during the generation process. Here, we follow [Mielke et al., 2022] and train an auxiliary classifier to predict model answer's correctness. For each instance, we compute the projection of the intermediate representation on the reflection direction across all layers. These values are concatenated into a feature vector p_{intv}:

$$\begin{aligned} p_{\text{intv}} &= \text{concat}(\{p_l^{\text{attn}}\}_{l=1}^{N_{\text{layer}}}, \{p_l^{\text{mlp}}\}_{l=1}^{N_{\text{layer}}}) \\ \text{where} \quad p_l^{\{\text{attn,mlp}\}} &= cos(d_l^{\{\text{attn,mlp}\}}, z_l^{\{\text{attn,mlp}\}}). \end{aligned} \tag{4}$$

We extract the feature vector p_{intv} from the end of thinking token (</think> for models we use in the paper), and train a logistic regression model upon it on GSM8k training dataset to predict whether model's answer is correct. If our hypothesis is correct, the classifier should achieve high accuracy, since the reflection direction is aligned with the model's uncertainty. As the baseline, we use the representation of the last token at the final layer. Results on the GSM8k test set (table 1) show that features derived from the reflection direction achieves higher AUROC and F1 scores, despite having fewer dimentions. This suggests that **model's uncertainty information is encoded in the reflection direction**, and may be a key factor in triggering self-reflection.

Model	final layer embedding		reflection direction	
	AUROC	F1	AUROC	F1
deepseek-llama-8b	0.736	0.946	**0.772**	**0.948**
qwq-32b	0.555	0.636	**0.564**	**0.839**
deepseek-qwen-14b	0.716	0.929	**0.850**	**0.976**

Table 1: **Probing results for uncertainty detection.** We train a logistic regression classifier to predict answers' correctness using (i) the last token embedding at the final layer or (ii) feature vector p_{intv} derived from reflection direction. Reflection-based features achieve higher AUROC and F1 scores despite lower dimensionality, suggesting that uncertainty is encoded in the reflection direction.



(a) Deepseek-r1 Llama 8b

(b) QwQ-32b

Figure 2: **Accuracy and number of reflection steps under different intervention strengths.** Results are shown for Deepseek-R1 Llama 8b (distilled model) and QwQ-32b (non-distilled model) on GSM8k. Accuracy remains largely stable, while the number of reflection steps decreases as intervention strength becomes more negative.

4 Experiments

In this section, we conduct an empirical study of our reflection extraction method.

4.1 Settings

Models. In this work, we mainly study the Deepseek-r1-distilled series of models, including the distilled version of Qwen-2.5 14B and Llama 8B, as these models are open-sourced. Additionally, we evaluate the QwQ-32B model as a non-distilled reasoning model.

Datasets. For math tasks, we use the GSM8k and MATH-500 as test datasets. For general reasoning tasks, we use the MMLU benchmark, selecting three subsets: Profeessional account, highschool computer science and formal logic.

Generation settings. We follow the standard generation configurations for each model. For math tasks, we use the prompt "Please reason step by step, and put your final answer within \boxed" after each question. For MMLU benchmark, we use the prompt ""Please reason step by step, and put your final answer (only the letter) within \boxed." The maximum completion tokens are set to 8192m, except for MATH-500 where we use 16384 due to its higher complexity.

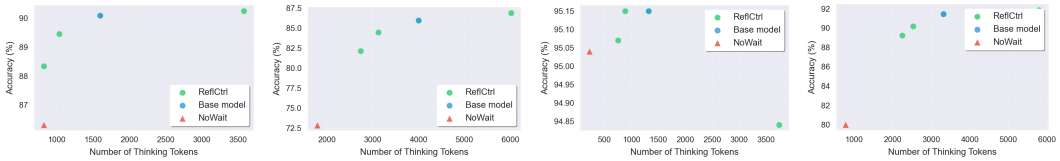
Reflection direction extraction. To extract the reflection direction, we use the GSM8k dataset to generate model responses. Then, we apply the method we propose in Sec. 3 to extract the direction. We omit the last step in the reasoning process as we observe that it is usually a conclusion sentence that is not related to reasoning.

Steering. For the results we show in this section, we apply the stepwise steering method we propose in Sec. 3 unless otherwise specified. The intervention is applied in all layers except the first and last six layers. We further discuss this choice in Sec. 4.4.

4.2 Main experiments

To answer **RQ2:** *How does reflection influence the model's reasoning performance*, we apply different strengths of interventions to the model to steer self-reflection, and check the model's performance change. Fig. 2 shows the accuracy and number of reflection steps under different intervention strengths. We choose Deepseek-R1 Llama 8b as the distilled model and QwQ-32b for non-distilled model. The results show that in both cases, the intervention effectively controls the number of reflection steps. In terms of accuracy, Deepseek-R1 Llama 8b gains marginal improvement with more reflections, while QwQ-32b is largely insensitive to the number of reflections. We observe:

1. Most models are less sensitive to additional reflections. From the table, the only model that benefits from positive intervention (more reflections) is the Deepseek-Llama-8b dis-



(a) Llama 8b, GSM8k (b) Llama 8b, MATH-500 (c) Qwen 14b, GSM8k (d) Qwen 14b, Math-500

Figure 3: **Accuracy versus reasoning token usage for ReflCtrl compared with NoWait [Wang et al., 2025a].** Results are shown for Deepseek-R1 Llama 8b and Deepseek-R1 Qwen 14b across GSM8k and MATH-500 benchmarks. ReflCtrl allows fine-grained control of the trade-off between accuracy and reasoning cost via intervention strength, while NoWait can only suppress reflections entirely. Additionally, ReflCtrl achieves lower performance loss for similar token usage.

tilled model, which receives 0.16% and 0.92% accuracy gain on GSM8k and MATH-500, respectively, at the cost of around 2000 additional reasoning tokens for each question.

2. Reflection redundancy exists in many cases, especially for stronger models. For example, in QwQ-32b model, the largest model in our test, the performance loss is only 0.14% and 0.34% on two datasets at intervention strength -0.96, while the reasoning token budget is reduced by 32.4% and 21.0%, respectively. This demonstrates that reasoning cost, which is substantially higher than non-reasoning models, can be reduced with minimal accuracy loss.

To further understand how reflection controls the trade-off between thinking cost and performance, we calculate the reasoning token usage and accuracy under different intervention strengths and report the results in table 2. For each question, we sample 10 responses and report the mean result. The results confirm our findings, showing that in many cases, reflections can be reduced without sacrificing performance.

To further understand the effectiveness of ReflCtrl, we compare it with the baseline, NoWait [Wang et al., 2025a]. NoWait is a recent work that reduces redundant reflection by directly suppressing corresponding reflection tokens. We plot the accuracy versus number of thinking tokens in Fig. 3. From the results, we can see that ReflCtrl is more flexible: the intervention strength can control the trade-off between performance and cost, while NoWait can only completely disable reflection. Additionally, ReflCtrl generally incurs smaller performance loss under similar token budget.

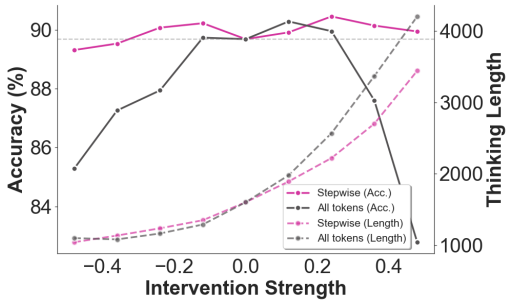
Category	Model	Metric	Reflection Strength			
			-0.96	-0.48	0	0.48
GSM-8k	DS-Llama-8b	Accuracy	88.34%	89.46%	90.09%	90.25%
		Tokens	821.0	1032.6	1595.7	3577.1
	QwQ-32b	Accuracy	96.36%	96.50%	96.50%	96.44%
		Tokens	1006.7	1162.5	1488.6	2256.9
	DS-qwen-14b	Accuracy	95.07%	95.15%	95.15%	94.84%
		Tokens	747.8	880.2	1315.9	3746.4
MATH-500	DS-Llama-8b	Accuracy	82.14%	84.46%	85.98%	86.90%
		Tokens	2738.1	3123.8	4000.7	6017.8
	QwQ-32b	Accuracy	92.72%	92.58%	93.06%	93.08%
		Tokens	2992.9	3253.4	3786.0	5028.9
	DS-qwen-14b	Accuracy	89.22%	90.18%	91.44%	91.86%
		Tokens	2247.1	2534.7	3315.3	5789.0

Table 2: **Accuracy and average reasoning token usage under different intervention strengths.** Results are reported on GSM8k and MATH-500 datasets for Deepseek-R1 Llama 8B, and Deepseek-R1 Qwen 14B, and QwQ-32B. Negative intervention strengths reduce reflection frequency and reasoning token usage with minimal accuracy loss, suggesting potential reflection redundancy.

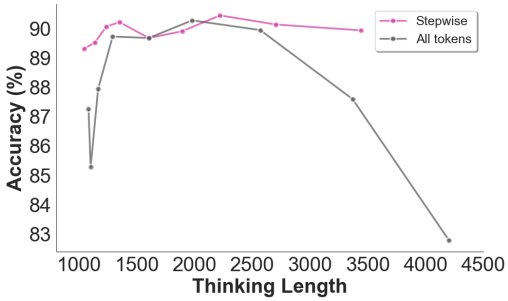
4.3 Stepwise steering

In this section, we study the stepwise steering strategy for controlling model reflection. We compare it with a baseline method where the intervention is applied to all generation tokens. In this experiment, we use Deepseek-R1 Llama 8b as the base model and evaluate on GSM8k dataset. As shown in figs. 4a and 4b, we observe that:

1. Under the same intervention strength, stepwise intervention achieves performance similar to intervention at all tokens. The baseline method produces stronger effects when applying positive intervention, i.e. increasing model's reflection.
2. In terms of accuracy, stepwise intervention maintains accuracy close to the original model, whereas the baseline method degrades performance significantly at larger intervention strengths (< -0.2 or > 0.3). Fig. 4b further shows that, under the same thinking token usage, stepwise intervention generally achieves higher accuracy.



(a) Accuracy and reasoning token usage under different intervention strength.



(b) Accuracy under different reasoning token usage.

Figure 4: **Comparison of stepwise versus all-token steering.** (a) Accuracy under different intervention strengths when interventions are applied at the start of each reasoning step (stepwise) or at every token (all-token). (b) Accuracy versus reasoning token usage under the two approaches. Stepwise steering preserves accuracy while reducing cost, whereas all-token steering causes significant degradation at large intervention strengths.

4.4 Abalation study: impact of layers

In this section, we study the effect of applying intervention at different layers of the LLM. We experiment with two strategies: Skipping the fisrt k layers and skipping the last k layers. The results in Fig. 5 indicate that the performance is best when skipping both the first and last six layers. We adopt this configuration accordingly in our main experiments.

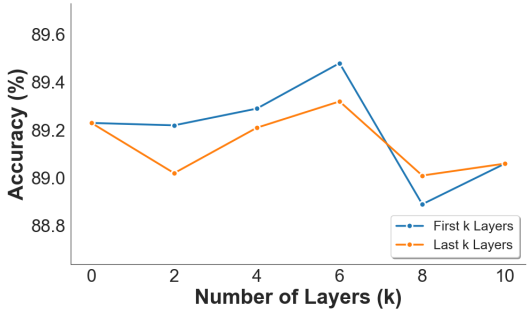


Figure 5: **Effect of applying interventions to different layers of the LLM.** We vary the number of skipped layers at the bottom and top of the network, with intervention strength fixed at -0.48. Accuracy is highest when skipping the first and last six layers, which we adopt as the default configuration in our main experiments.

4.5 Non-math benchmarks

To evaluate the proposed ReflCtrl on non-math tasks, we conduct extensive experiments on non-math datasets. As shown in table 3, we observe a similar phenomenon as in math tasks. In general, the smallest model, Deepseek Llama 8b, is most sensitive to reflection. In contrast, the larger Deepseek Qwen 14b and QwQ-32b are hardly affected by the reduction of reflections. Using the proposed stepwise steering, up to 33.6% of reasoning tokens can be saved.

Category	Model	Metric	Reflection Strength			
			-0.96	-0.48	0	0.48
Professional accounting	DS-Llama-8b	Accuracy	50.1%	53.4%	56.5%	57.3%
		Tokens	1453.6	1668.8	2097.5	2807.7
	DS-qwen-14b	Accuracy	78.5%	76.8%	77.8%	77.6%
		Tokens	983.9	1103.1	1482.1	2470.1
	QwQ-32b	Accuracy	89.3%	89.5%	88.5%	89.2%
		Tokens	1231.2	1313.7	1648.0	2234.3
Highschool computer science	DS-Llama-8b	Accuracy	79.6%	82.7%	87.3%	88.0%
		Tokens	1016.1	1157.9	1365.4	1970.4
	DS-qwen-14b	Accuracy	95.2%	95.4%	95.0%	94.8%
		Tokens	711.9	787.9	933.5	1498.7
	QwQ-32b	Accuracy	96.6%	96.2%	96.7%	97.0%
		Tokens	771.6	741.9	871.0	1004.7
Formal logic	DS-Llama-8b	Accuracy	60.5%	61.0%	62.1%	62.7%
		Tokens	2266.5	2586.9	3378.3	4553.5
	DS-qwen-14b	Accuracy	91.8%	92.2%	92.6%	92.8%
		Tokens	1287.2	1440.1	1891.4	3196.5
	QwQ-32b	Accuracy	96.3%	95.5%	95.7%	96.0%
		Tokens	1481.4	1447.8	1716.6	2175.6

Table 3: **Accuracy and reasoning token usage under different reflection strengths on MMLU subsets.** Smaller models (e.g., DS-Llama-8B) are more sensitive to reflection reduction, while larger models (DS-Qwen-14B and QwQ-32B) maintain accuracy with fewer reflections, saving up to 33.6% of reasoning tokens.

5 Conclusion and limitation

In this work, we propose ReflCtrl, a representation engineering framework for understanding and steering self-reflection behavior in reasoning LLMs. By segmenting model reasoning into thinking steps and identifying reflection-related steps, we extract a reflection direction in the latent space, enabling direct control over the number of self-reflections produced during inference. We further introduce a **stepwise steering** strategy that only applies interventions at the start of new thinking steps, substantially reducing reasoning token usage while preserving performance. Across multiple math and general-domain reasoning benchmarks, we find that:

1. Reflection redundancy is common, particularly in stronger models where minimal accuracy loss is observed when reflections are reduced.
2. Reflection direction is correlated with internal uncertainty signals, implying that the reflection behavior may be controlled by model's internal uncertainty perception.
3. Stepwise steering can largely mitigate performance loss. Compared with regular intervention, stepwise steering can achieve similar intervention performance with over 5% accuracy improvement.

Despite these promising results, our work also has some limitations. First, the identification of reasoning steps relies on keyword search, which may be model specific since different models could prefer different reflection cues. Second, our ReflCtrl only works for open-source models and it remains unclear how it works for SOTA close-source models such as GPT-4 or Claude, which is a shared limitation of representation engineering methods.

For future work, we believe that developing uncertainty-aware dynamic steering is a promising direction: our results have preliminarily shown a connection between uncertainty and self-reflection. The current steering method applies a fixed strength across all questions and throughout the generation process. Enabling the model to dynamically adjust steering strength during inference could substantially improve reflection efficiency and further reduce inference cost in reasoning LLMs.

References

Lukasz Bartoszcze, Sarthak Munshi, Bryan Sukidi, Jennifer Yen, Zejia Yang, David Williams-King, Linh Le, Kosi Asuzu, and Carsten Maple. Representation engineering for large-language models: Survey and research challenges. *arXiv preprint arXiv:2502.17601*, 2025.

DeepSeekAI, Daming Guo, Dayi Yang, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

Sabrina J. Mielke, Arthur Szlam, Emily Dinan, and Y-Lan Boureau. Reducing conversational agents' overconfidence through linguistic calibration. *Transactions of the Association for Computational Linguistics*, 10:857–872, 2022. doi: 10.1162/tacl_a_00494. URL https://aclanthology.org/2022.tacl-1.50/.

OpenAI. Openai o1 system card, 2024.

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

Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown, November 2024. URL https://qwenlm.github.io/blog/qwq-32b-preview/.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Chenlong Wang, Yuanning Feng, Dongping Chen, Zhaoyang Chu, Ranjay Krishna, and Tianyi Zhou. Wait, we don't need to" wait"! removing thinking tokens improves reasoning efficiency. *arXiv preprint arXiv:2506.08343*, 2025a.

Mengru Wang, Xingyu Chen, Yue Wang, Zhiwei He, Jiahao Xu, Tian Liang, Qiuzhi Liu, Yunzhi Yao, Wenxuan Wang, Ruotian Ma, et al. Two experts are all you need for steering thinking: Reinforcing cognitive effort in moe reasoning models without additional training. *arXiv preprint arXiv:2505.14681*, 2025b.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022.

Shu Yang, Junchao Wu, Xin Chen, Yunze Xiao, Xinyi Yang, Derek F Wong, and Di Wang. Understanding aha moments: from external observations to internal mechanisms. *arXiv preprint arXiv:2504.02956*, 2025.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023.