
ThinkEdit: Interpretable Weight Editing to Mitigate Overly Short Thinking in Reasoning Models

Chung-En Sun
UCSD CSE
cesun@ucsd.edu

Ge Yan
UCSD CSE
geyan@ucsd.edu

Tsui-Wei Weng
UCSD HDSI
lweng@ucsd.edu

Abstract

Recent studies have shown that Large Language Models (LLMs) augmented with chain-of-thought (CoT) reasoning demonstrate impressive problem-solving abilities. However, in this work, we identify a recurring issue where these models occasionally generate overly short reasoning, leading to degraded performance on even simple mathematical problems. Specifically, we investigate how reasoning length is embedded in the hidden representations of reasoning models. Our analysis reveals that reasoning length is governed by a linear direction in the representation space, allowing us to induce overly short reasoning by steering the model along this direction. Building on this insight, we introduce *ThinkEdit*, an effective weight-editing approach to mitigate the issue of overly short reasoning. We first identify a small subset of attention heads (approximately 4%) that predominantly drive short reasoning behavior, and then edit the output projection weights of these heads to remove the short reasoning direction. With changes to only 0.2% of the parameters, *ThinkEdit* effectively reduces overly short reasoning and yields notable accuracy gains for short reasoning outputs (+6.39%), along with an overall improvement (+3.34%) across multiple math benchmarks. Our code is available at: <https://github.com/Trustworthy-ML-Lab/ThinkEdit>

1 Introduction

Reinforcement Learning (RL) has strengthened Large Language Models (LLMs) with chain-of-thought (CoT) reasoning Jaech et al. [2024], Guo et al. [2025], Muennighoff et al. [2025]. Despite these advances, performance still falls short of perfect accuracy on benchmarks like GSM8K Cobbe et al. [2021]. As shown in Section 2, Deepseek-distilled reasoning models occasionally produce overly short reasoning chains, and such cases have about 20% lower accuracy on MATH-level5 Hendrycks et al. [2021b]. This pattern appears consistently across model sizes, suggesting that reasoning length is a key factor in problem-solving, yet its internal control mechanisms remain underexplored.

To bridge this gap, we examine how reasoning length is represented in a model’s hidden states and extract a *reasoning length direction*—a latent feature in the residual stream that enables direct control over length (Figure 1, left). We find that overly short, abstract reasoning degrades performance and is mainly encoded in the middle layers. Moreover, about 4% of attention heads in these layers disproportionately drive short reasoning. Building on these insights, we propose *ThinkEdit*, an effective weight-editing technique that removes the short-reasoning component from these heads (Figure 1, right), boosting both short-case and overall accuracy. Our contributions are:

- Identify the prevalence and impact of overly short reasoning in Deepseek-distilled models.
- Extract a latent *reasoning length direction*, revealing middle layers’ role in length control.
- Discover and edit a small set of “short reasoning” heads using *ThinkEdit*, improving short reasoning accuracy (+6.39%) and overall accuracy (+3.34%).

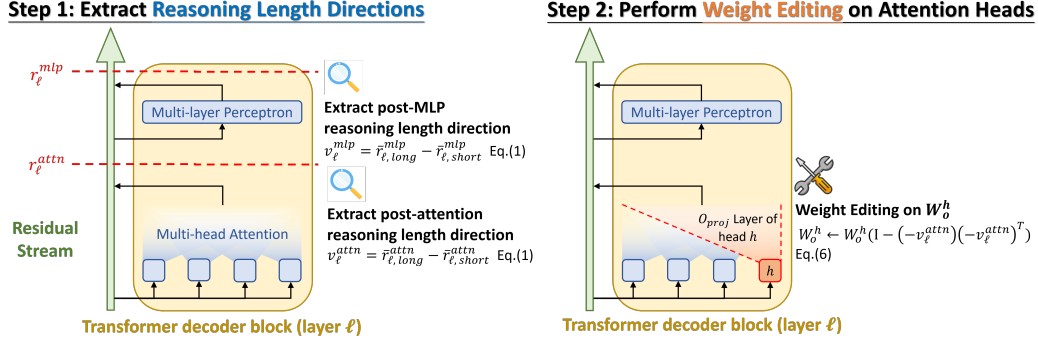


Figure 1: The overview of **ThinkEdit** framework. We first identify linear directions for controlling reasoning length in the hidden states, and then perform weight editing on the key attention heads.

Due to space limitations, a detailed discussion of related works is deferred to Appendix A.1.

2 Unexpectedly Low Accuracy in Short Reasoning Cases

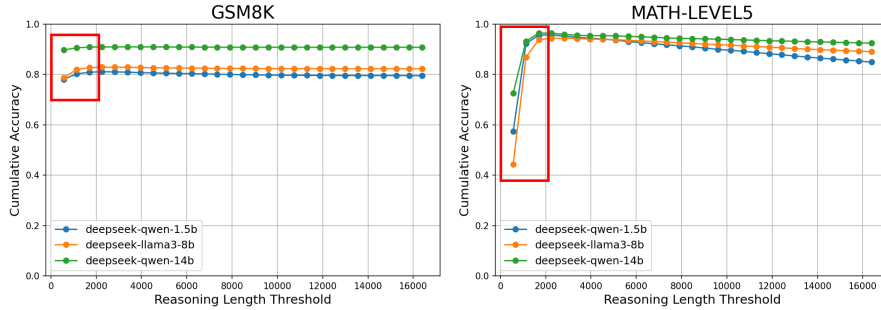


Figure 2: Cumulative accuracy versus reasoning length threshold, showing average accuracy for responses below each cutoff. Accuracy drops markedly for short reasoning (e.g., <1000 tokens).

Across datasets such as GSM8K Cobbe et al. [2021] and MATH-Level5 Hendrycks et al. [2021b], Deepseek-distilled reasoning models show markedly lower accuracy when reasoning is short. Figure 2 plots cumulative accuracy (y-axis) against a cutoff on reasoning length (x-axis); for example, a cutoff of 2000 means averaging accuracy over all responses with at most 2000 tokens. The results reveal a sharp and unexpected drop in performance for short reasoning, indicating that brief chains of thought are a consistent failure mode. Motivated by this, we examine how internal representations control reasoning length and its effect on accuracy (Section 3), and later introduce **ThinkEdit** (Section 4) to address this issue by editing the output layers of a few key attention heads.

3 Understanding How Representations Affect Reasoning Length

In this section, we study how reasoning length is encoded in hidden states and how manipulating it affects performance.

3.1 Extracting Reasoning Length Directions

To study how *reasoning length* is encoded in a model’s hidden states, we collect responses to 2,000 GSM8K Cobbe et al. [2021] training problems, where the chain-of-thought (CoT) is enclosed by `<think>...</think>`. Reasoning length is measured by counting only tokens within these tags. We build two datasets: $\mathcal{D}_{\text{long}}$ (CoT > 1000 tokens) and $\mathcal{D}_{\text{short}}$ (CoT < 100 tokens). Each entry contains: (1) the problem, (2) the extracted CoT, enclosed by the tags, and (3) the step-by-step solution.

We input the problem and CoT into the model and extract *post-attention* and *post-MLP* residual stream states, r_{ℓ}^{attn} and r_{ℓ}^{mlp} , at each layer ℓ . Let $r_{\ell}^x(i, t)$ be the hidden state for token t in problem i , with $x \in \{\text{attn}, \text{mlp}\}$, and $\mathcal{T}_i$ the token positions within the CoT. We compute the mean CoT

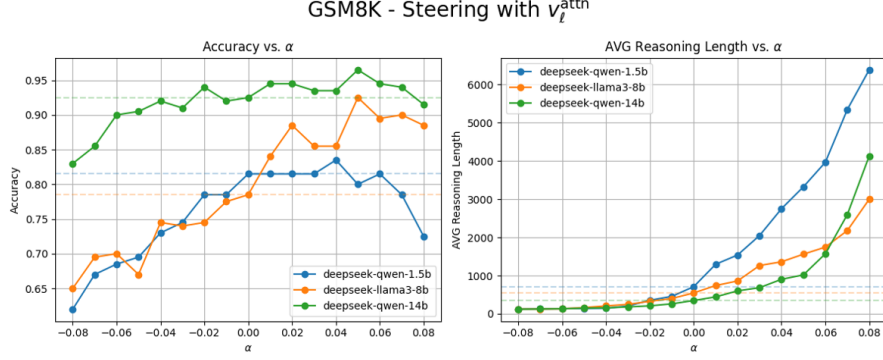


Figure 3: Steering results. Positive α extends reasoning length and improves accuracy in the 8B and 14B models, while negative α shortens reasoning and lowers accuracy.

representation for each problem and then average over all problems in $\mathcal{D}_y$ ($y \in \{\text{long}, \text{short}\}$): $\bar{r}_{\ell,y}^x = \frac{1}{|\mathcal{D}_y|} \sum_{i \in \mathcal{D}_y} \frac{1}{|\mathcal{T}_i|} \sum_{t \in \mathcal{T}_i} r_{\ell}^x(i, t)$. The *reasoning-length direction* at layer ℓ is then:

$$v_{\ell}^{\text{attn}} = \bar{r}_{\ell,\text{long}}^{\text{attn}} - \bar{r}_{\ell,\text{short}}^{\text{attn}}, \quad v_{\ell}^{\text{mlp}} = \bar{r}_{\ell,\text{long}}^{\text{mlp}} - \bar{r}_{\ell,\text{short}}^{\text{mlp}}. \quad (1)$$

These vectors capture representation differences between long and short reasoning, which we later manipulate to study their effect on reasoning length and accuracy.

3.2 Effects of Reasoning-Length Direction

We steer models by shifting residual states:

$$r_{\ell}^{\text{attn}} \leftarrow r_{\ell}^{\text{attn}} + \alpha v_{\ell}^{\text{attn}}, \quad r_{\ell}^{\text{mlp}} \leftarrow r_{\ell}^{\text{mlp}} + \alpha v_{\ell}^{\text{mlp}}, \quad (2)$$

where $\alpha \in [-0.08, 0.08]$; $\alpha > 0$ promotes longer reasoning, $\alpha < 0$ shorter reasoning.

Steering Results and Key Insights. We evaluate on GSM8K (200 problems) using deepseek-distill-{qwen-1.5B, llama3-8B, qwen-14B}. As shown in Figure 3, increasing α lengthens the CoT and boosts accuracy for the 8B/14B models (up to +10%) but reduces it for the 1.5B model. Layerwise steering (Appendix A.3) further shows that **middle layers** exert the strongest control over reasoning length. Notably, steering toward shorter reasoning ($\alpha < 0$) consistently degrades accuracy, revealing a distinct short-reasoning pattern in hidden states. These findings motivate identifying and editing key components within the middle layers (Section 4).

4 ThinkEdit: Mitigating Overly Short Reasoning via Weight Editing

Building on Section 3.2, we propose **ThinkEdit**, a targeted weight-editing method to mitigate overly short reasoning. We focus on attention heads, identifying those most aligned with the short-reasoning direction and removing this component from their outputs.

4.1 Identifying and Editing Short Reasoning Heads

We begin by defining each head’s contribution to the residual stream. In a multi-head attention layer, head h writes $C^h = A^h W_o^h \in \mathbb{R}^{T \times d}$ to the residual stream, where A^h is the attention-weighted value matrix and W_o^h is the output projection. We can view C^h as the change contributed by head h to the residual stream.

For each short-reasoning example in $\mathcal{D}_{\text{short}}$, we average C^h over all chain-of-thought token positions $\mathcal{T}_i$: $\bar{C}^h = \frac{1}{|\mathcal{D}_{\text{short}}|} \sum_{i \in \mathcal{D}_{\text{short}}} \frac{1}{|\mathcal{T}_i|} \sum_{t \in \mathcal{T}_i} C^h(i, t)$. We then project $\bar{C}^h$ onto the short-reasoning direction $-\hat{v}_{\ell}^{\text{attn}}$ (Eq. 1) to measure its alignment: $\bar{C}_{\text{short}}^h = \langle \bar{C}^h, -\hat{v}_{\ell}^{\text{attn}} \rangle$. Heads with large $\bar{C}_{\text{short}}^h$ are identified as *short reasoning heads*. Figure 4 shows that such heads are sparse and concentrated in the middle layers, consistent with Section 3.2.

We select the top 4% of heads by $\bar{C}_{\text{short}}^h$ and edit their output projection matrices $W_o^{h_{\ell}}$ to remove the short-reasoning component:

Short Reasoning Attention Head Distribution

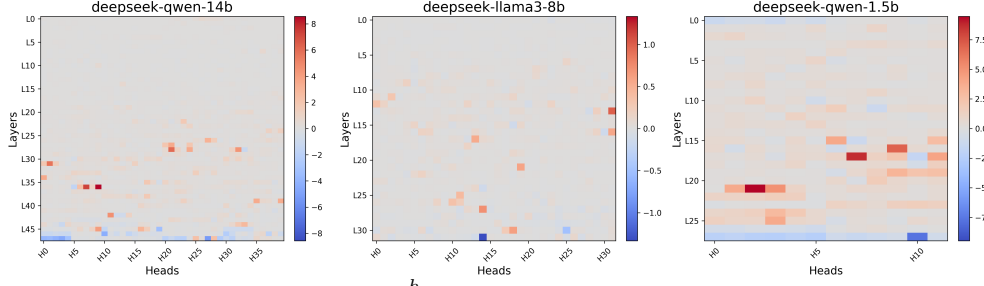


Figure 4: Short reasoning contribution $\bar{C}_{\text{short}}^h$. Red indicates stronger alignment with short reasoning.

Model		GSM8K	MMLU Elem. Math	MATH-Level1	MATH-Level5	MATH-500
deepseek-qwen-14B	Original	90.80 $\pm$ 0.36	95.08 $\pm$ 0.65	96.32 $\pm$ 0.35	90.25 $\pm$ 0.72	91.48 $\pm$ 0.55
	ThinkEdit (4%)	93.78 $\pm$ 0.50	96.56 $\pm$ 0.84	96.36 $\pm$ 0.52	91.03 $\pm$ 0.44	91.92 $\pm$ 0.63
deepseek-llama3-8B	Original	82.26 $\pm$ 0.91	96.01 $\pm$ 0.62	93.46 $\pm$ 0.84	85.49 $\pm$ 0.83	87.26 $\pm$ 1.16
	ThinkEdit (4%)	89.44 $\pm$ 0.55	96.19 $\pm$ 0.73	94.44 $\pm$ 0.31	86.49 $\pm$ 0.54	88.06 $\pm$ 1.09
deepseek-qwen-1.5B	Original	79.15 $\pm$ 1.08	68.52 $\pm$ 1.56	93.00 $\pm$ 0.33	75.48 $\pm$ 0.90	82.22 $\pm$ 1.29
	ThinkEdit (4%)	84.56 $\pm$ 0.79	90.66 $\pm$ 0.97	93.66 $\pm$ 0.62	75.05 $\pm$ 0.82	82.24 $\pm$ 0.89

Table 1: Overall accuracy (%) of each model before and after applying ThinkEdit.

Model		GSM8K	MMLU Elem. Math	MATH-Level1	MATH-Level5	MATH-500
deepseek-qwen-14b	Original	96.31 / 95.65 / 92.93	93.89 / 96.22 / 95.60	99.52 / 99.30 / 97.70	89.39 / 94.32 / 96.25	86.40 / 91.40 / 93.50
	ThinkEdit (4%)	96.31 / 96.18 / 96.77	97.78 / 95.14 / 96.53	99.52 / 99.53 / 98.62	96.67 / 97.88 / 98.11	91.20 / 93.20 / 95.00
deepseek-llama3-8b	Original	88.92 / 87.18 / 85.82	97.22 / 96.49 / 96.80	97.14 / 94.88 / 94.83	78.64 / 88.79 / 93.41	82.00 / 81.40 / 88.30
	ThinkEdit (4%)	96.31 / 95.50 / 94.68	97.78 / 97.57 / 97.60	99.05 / 99.07 / 97.82	95.76 / 97.42 / 97.46	95.60 / 93.80 / 95.40
deepseek-qwen-1.5b	Original	88.46 / 87.48 / 85.02	62.78 / 62.16 / 60.53	97.62 / 95.12 / 93.91	91.52 / 95.00 / 95.72	82.40 / 89.80 / 93.40
	ThinkEdit (4%)	92.62 / 92.90 / 92.32	87.78 / 88.11 / 88.67	95.71 / 95.58 / 96.44	95.15 / 96.59 / 97.27	90.80 / 92.00 / 94.20

Table 2: Accuracy (%) of the top 5% / 10% / 20% shortest reasoning responses.

$$W_o^{h_\ell} \leftarrow W_o^{h_\ell} (I - (-\hat{v}_\ell^{\text{attn}})(-\hat{v}_\ell^{\text{attn}})^\top).$$

This projects $W_o^{h_\ell}$ orthogonally to $-\hat{v}_\ell^{\text{attn}}$, removing the short-reasoning direction from the head’s output. Since we only modify the output projection W_o of each selected head, editing 4% of heads changes just 0.2% of the model’s total parameters.

4.2 Performance After *ThinkEdit*

Experimental Setup. We evaluate on GSM8K Cobbe et al. [2021], MMLU Elementary Math Hendrycks et al. [2021a], MATH-Level1 and MATH-Level5 Hendrycks et al. [2021b], and MATH-500 Lightman et al. [2023] averaged over 10 runs.

Overall and Short Reasoning Accuracy. Table 1 shows consistent gains across all datasets, with large improvements on simpler tasks (e.g., MMLU Elementary Math) and smaller but still positive gains on harder benchmarks. Furthermore, Table 2 demonstrates notable improvements in the top 5%, 10%, and 20% shortest reasoning cases, including challenging benchmarks such as MATH-Level5 and MATH-500, indicating that *ThinkEdit* effectively mitigates the accuracy drop associated with brief reasoning.

5 Conclusion

We identify overly short reasoning as a common failure mode in Deepseek-distilled reasoning models and trace it to a latent direction in hidden representations. By locating the 4% of attention heads driving this behavior, we introduce *ThinkEdit*, which boosts short reasoning accuracy by +6.39% and overall accuracy by +3.34% across multiple math benchmarks.

References

- Jianhui Chen, Xiaozhi Wang, Zijun Yao, Yushi Bai, Lei Hou, and Juanzi Li. Finding safety neurons in large language models. *CoRR*, 2024.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, 2021.
- Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. In *ICLR*, 2020.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv*, 2025.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *ICLR*, 2021a.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021b.
- Zheng Yi Ho, Siyuan Liang, Sen Zhang, Yibing Zhan, and Dacheng Tao. Novo: Norm voting off hallucinations with attention heads in large language models. *ICLR*, 2025.
- Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Hel-
yar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Pas-
sos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Ku-
mar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko,
Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Ben Rossen, Ben-
jamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon
Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson,
Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris
Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel
Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras,
Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch
Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such,
Filippo Raso, Florencia Leoni, Foivos Tsimpouras, Francis Song, Fred von Lohmann, Freddie
Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman,
Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad,
Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O’Connell, Ian Osband,
Ignasi Clavera Gilaberte, and Ilge Akkaya. Openai o1 system card. *CoRR*, 2024.
- Kai Konen, Sophie Jentzsch, Diaoulé Diallo, Peer Schütt, Oliver Bensch, Roxanne El Baff, Dominik
Opitz, and Tobias Hecking. Style vectors for steering generative large language models. In Yvette
Graham and Matthew Purver, editors, *EACL Findings*, 2024.
- Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In
Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *ACL/IJCNLP*, 2021.
- Xun Liang, Hanyu Wang, Yezhaohui Wang, Shichao Song, Jiawei Yang, Simin Niu, Jie Hu, Dan Liu,
Shunyu Yao, Feiyu Xiong, and Zhiyu Li. Controllable text generation for large language models:
A survey. *CoRR*, 2024.
- Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan
Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step. *arXiv preprint
arXiv:2305.20050*, 2023.
- Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. Autodan: Generating stealthy jailbreak
prompts on aligned large language models. In *ICLR*, 2024.

- Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *NeurIPS*, 2022.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *NeurIPS*, 2023.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, 2024.
- Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *EMNLP*, 2020.
- Nishant Subramani, Nivedita Suresh, and Matthew E. Peters. Extracting latent steering vectors from pretrained language models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *ACL Findings*, 2022.
- Chung-En Sun, Xiaodong Liu, Weiwei Yang, Tsui-Wei Weng, Hao Cheng, Aidan San, Michel Galley, and Jianfeng Gao. Iterative self-tuning llms for enhanced jailbreaking capabilities. *arXiv preprint arXiv:2410.18469*, 2024a.
- Chung-En Sun, Tuomas Oikarinen, Berk Ustun, and Tsui-Wei Weng. Concept bottleneck large language models. *arXiv preprint arXiv:2412.07992*, 2024b.
- Chung-En Sun, Tuomas Oikarinen, and Tsui-Wei Weng. Crafting large language models for enhanced interpretability. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024c.
- Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. In *ICLR*, 2022.
- Lei Yu, Meng Cao, Jackie C. K. Cheung, and Yue Dong. Mechanistic understanding and mitigation of language model non-factual hallucinations. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *EMNLP Findings*, 2024.
- Yoel Zeldes, Dan Padnos, Or Sharir, and Barak Peleg. Technical report: Auxiliary tuning and its application to conditional text generation. *CoRR*, 2020.
- Yiran Zhao, Wenxuan Zhang, Yuxi Xie, Anirudh Goyal, Kenji Kawaguchi, and Michael Shieh. Understanding and enhancing safety mechanisms of llms via safety-specific neuron. *ICLR*, 2025.
- Zhenhong Zhou, Haiyang Yu, Xinghua Zhang, Rongwu Xu, Fei Huang, Kun Wang, Yang Liu, Junfeng Fang, and Yongbin Li. On the role of attention heads in large language model safety. *ICLR*, 2025.
- Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to AI transparency. *CoRR*, 2023a.
- Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *CoRR*, 2023b.

A Appendix

A.1 Related Works

Reasoning Models. Recent advances in reasoning models have significantly improved the problem-solving abilities of LLMs in domains such as mathematics, coding, and science. OpenAI’s o1 Jaech et al. [2024] represents a major shift toward deliberate reasoning by employing reinforcement learning (RL) to refine its strategies. By generating explicit "Thinking" steps before producing answers, o1 achieves strong performance on complex tasks. As a more cost-efficient alternative, DeepSeek-r1 Guo et al. [2025] demonstrates that pure RL can also effectively enhance reasoning. It introduces Group Relative Policy Optimization (GRPO) Shao et al. [2024], a novel method that eliminates the need for a separate reward model, enabling more efficient RL training.

Controllable Text Generation. Controllable text generation has been explored across various domains Liang et al. [2024], with methods generally classified into training-time and inference-time control. These approaches aim to steer LLMs toward generating text with specific attributes while preserving fluency and coherence. Training-time control is achieved through fine-tuning Zeldes et al. [2020], Wei et al. [2022] or reinforcement learning Ouyang et al. [2022], Rafailov et al. [2023], leveraging diverse datasets to shape model behavior. Inference-time control encompasses prompt engineering Shin et al. [2020], Li and Liang [2021], representation engineering Subramani et al. [2022], Zou et al. [2023a], Konen et al. [2024], interpretable neuron intervention Sun et al. [2024b,c], and decoding-time interventions Dathathri et al. [2020], allowing flexible and efficient control without retraining.

In this work, we focus on the representation engineering paradigm to investigate how reasoning length is embedded within model representations. Specifically, we introduce a linear "reasoning-length direction" in the representation space to examine its impact on reasoning capabilities.

Attention heads and MLP neurons intervention. A growing body of research explores the role of attention heads and neurons within the Multi-Layer Perceptron (MLP) layers in shaping language model behavior. Studies such as Zhou et al. [2025], Zhao et al. [2025], Chen et al. [2024] examine how safety mechanisms are embedded in well-aligned models to defend against harmful prompts and jailbreak attacks Zou et al. [2023b], Liu et al. [2024], Sun et al. [2024a]. Findings indicate that a small subset of attention heads and MLP neurons play a critical role in safety alignments. Similarly, research on hallucination mitigation has investigated the contributions of attention heads and MLP neurons. Ho et al. [2025] demonstrates that filtering out unreliable attention heads can reduce erroneous generations, while Yu et al. [2024] finds that activating subject-knowledge neurons helps maintain factual consistency.

In our work, we investigate how attention heads relate to reasoning processes, examining their impact on the reasoning length and quality.

A.2 Global Steering with the MLP-based Direction v_{ℓ}^{mlp}

Figure 5 replicates the global steering analysis using the MLP-based direction v_{ℓ}^{mlp} . The observed trends closely mirror those from attention-based steering: increasing α extends reasoning length across both datasets, and the effect on accuracy is model- and dataset-dependent. On GSM8K, larger models benefit from longer reasoning, while smaller models degrade. On MATH-Level5, overly long reasoning may hurt performance, despite consistent control over CoT length. These results show that both attention and MLP directions capture similar reasoning-length attributes.

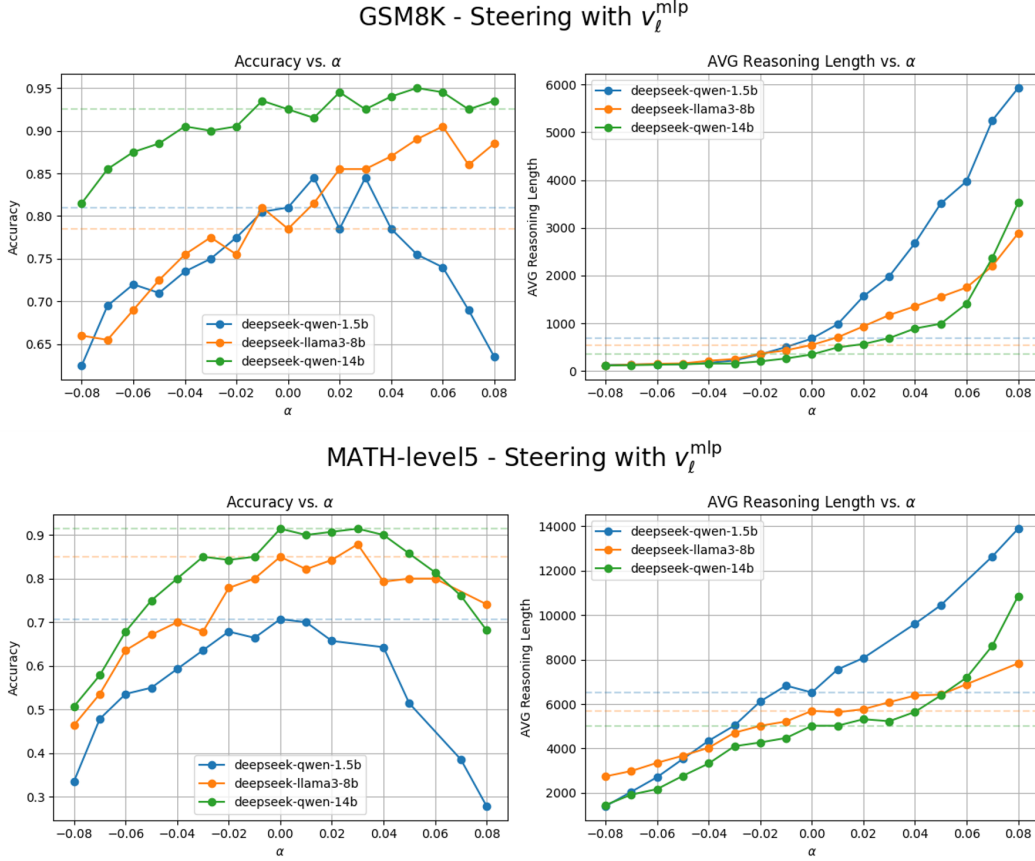


Figure 5: Global steering with the reasoning length direction extracted from MLPs. The trend is similar as steering with attention-based directions.

A.3 Layerwise Analysis of Steering along Reasoning Length Direction

To identify which layers are most influenced by the reasoning-length direction, we perform a *layerwise* experiment on the GSM8K dataset (Figure 6). Specifically, we use v_ℓ^{mlp} to steer each MLP layer *separately*, applying $\alpha = \pm 1$ at a single layer ℓ . Notably, the *middle layers* elicit the largest fluctuations, suggesting they encode key representations for controlling reasoning length.

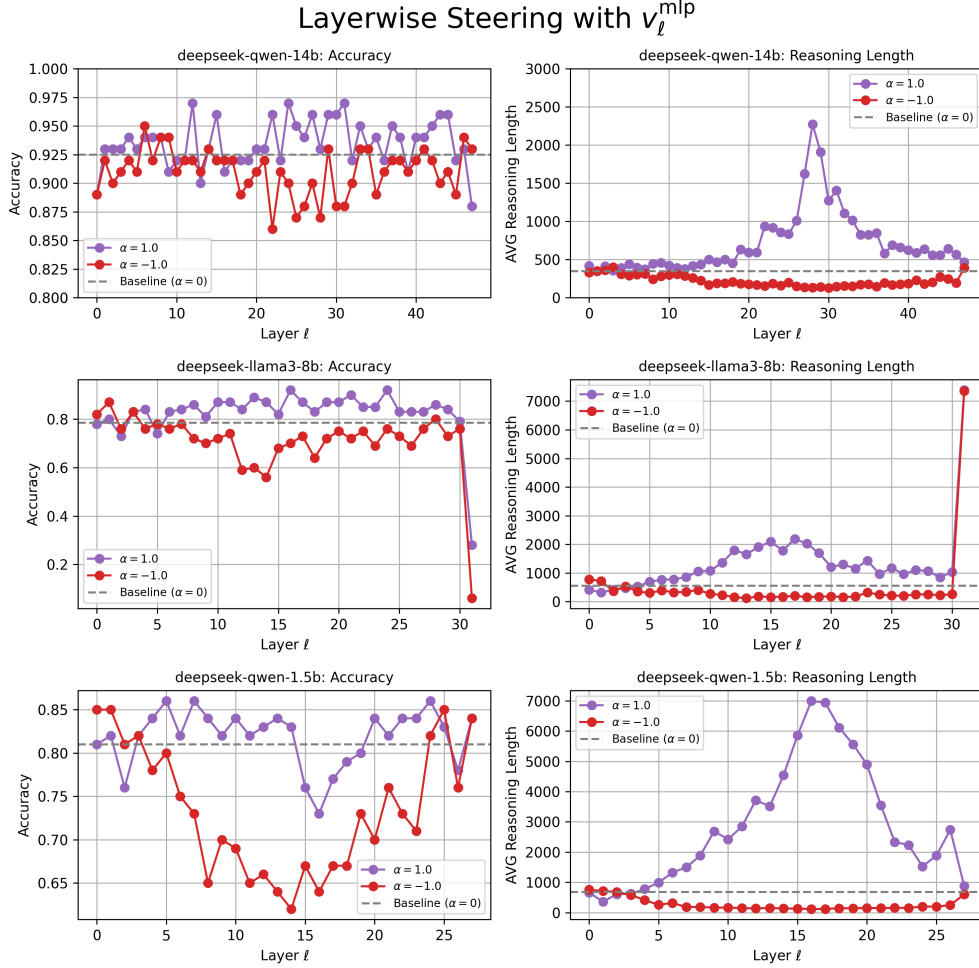


Figure 6: Layerwise steering on GSM8K with v_ℓ^{mlp} . We apply $\alpha = \pm 1$ to one layer at a time, revealing that the middle layers wield the strongest control over reasoning length and accuracy.

A.4 The Impact of ThinkEdit on Reasoning Length

Table 3 reports the average reasoning length among the top 5%, 10%, and 20% shortest responses. We observe that *ThinkEdit* consistently increases the reasoning length in these short-answer scenarios, suggesting that the intervention discourages excessively short reasoning, and therefore leads to higher accuracy as shown in Table 2. Interestingly, Table 4 shows that the average reasoning length remains similar between the original and *ThinkEdit* models. To summarize these trends, we compute the average change in reasoning length across all datasets: +2.94% for deepseek-qwen-14b, +3.53% for deepseek-llama3-8b, and -5.73% for deepseek-qwen-1.5b, resulting in an overall average change of -0.27%. These results suggest that *ThinkEdit* selectively increases reasoning length for short responses without significantly altering overall response length.

Model		GSM8K	MMLU Elem. Math	MATH-Level1	MATH-Level5	MATH-500
deepseek-qwen-14B	Original	76.6 / 86.5 / 99.1	65.8 / 72.2 / 80.6	93.7 / 114.3 / 188.6	628.8 / 858.4 / 1125.9	198.7 / 434.3 / 697.0
	ThinkEdit (4%)	101.7 / 113.6 / 131.0	82.7 / 91.8 / 105.6	146.7 / 188.6 / 346.0	745.5 / 926.6 / 1163.7	361.3 / 559.3 / 764.6
deepseek-llama3-8B	Original	73.0 / 83.1 / 96.6	371.0 / 438.1 / 518.2	80.3 / 97.2 / 130.3	617.9 / 854.9 / 1126.5	159.5 / 357.5 / 644.5
	ThinkEdit (4%)	110.3 / 131.8 / 164.6	398.5 / 462.4 / 541.8	176.3 / 221.3 / 336.0	806.1 / 963.3 / 1185.1	372.5 / 553.2 / 772.9
deepseek-qwen-1.5B	Original	78.8 / 89.4 / 103.0	61.6 / 68.5 / 77.6	88.8 / 110.3 / 219.7	804.6 / 1017.9 / 1314.0	249.7 / 506.5 / 760.7
	ThinkEdit (4%)	103.3 / 118.9 / 144.8	80.6 / 92.5 / 112.9	172.7 / 336.9 / 543.6	853.0 / 1003.5 / 1221.9	530.8 / 676.0 / 837.4

Table 3: Average reasoning length for the top 5% / 10% / 20% shortest responses (in tokens).

Model		GSM8K	MMLU Elem. Math	MATH-Level1	MATH-Level5	MATH-500
deepseek-qwen-14B	Original	354.5 ± 684.4	184.9 ± 175.3	1600.5 ± 1885.2	4306.2 ± 3816.1	3096.8 ± 3308.0
	ThinkEdit (4%)	538.2 ± 829.6	291.4 ± 607.5	1670.4 ± 1951.2	4243.7 ± 3814.0	3079.7 ± 3276.6
deepseek-llama3-8B	Original	597.3 ± 1109.0	1486.6 ± 2036.7	1646.6 ± 2275.0	4789.1 ± 4315.4	3507.6 ± 3917.5
	ThinkEdit (4%)	927.7 ± 1486.3	1517.9 ± 2041.5	1723.7 ± 2152.3	4773.5 ± 4327.4	3509.5 ± 3842.9
deepseek-qwen-1.5B	Original	768.1 ± 1837.2	517.0 ± 1502.8	2080.9 ± 2740.5	6360.0 ± 5336.4	4260.3 ± 4668.2
	ThinkEdit (4%)	1126.6 ± 2018.0	768.9 ± 1651.4	1946.3 ± 2438.4	5522.4 ± 5036.9	3821.1 ± 4384.9

Table 4: Overall reasoning length (in tokens) before and after applying ThinkEdit (4% edit rate).

A.5 ThinkEdit with Varying Editing Rates vs. the "Wait" Appending Baseline

We conduct a comprehensive evaluation of *ThinkEdit* with different editing rates and compare it against a simple baseline that appends the word "Wait" to reasoning sequences shorter than 500 tokens. This baseline is intended to prompt the model to continue thinking before answering when the reasoning is too short. The methods compared are:

- *ThinkEdit* (8%): Edits 8% of total attention heads.
- *ThinkEdit* (4%): Edits 4% of total attention heads.
- *ThinkEdit* (2%): Edits 2% of total attention heads.
- **Append "Wait"**: Appends "Wait" to reasoning with fewer than 500 tokens to artificially extend reasoning length.
- **Original**: The unmodified model output.

As shown in Table 5, higher editing rates in *ThinkEdit* consistently improve performance on GSM8K and MMLU Elementary Math. However, for the MATH-series datasets, moderate editing rates yield better results than the most aggressive edits. The "Append Wait" baseline yields only marginal gains across all datasets, indicating that simply forcing the model to produce longer reasoning does not necessarily improve accuracy. A closer look at the short reasoning cases in Table 7 shows that all versions of *ThinkEdit* substantially outperform the "Append Wait" baseline. This further supports the observation that longer reasoning alone is insufficient without proper internal adjustment of the model.

In terms of reasoning length (Tables 6 and 8), the "Append Wait" method generally leads to longer outputs than *ThinkEdit* (2%), confirming that it effectively increases response length. However, despite this, it fails to match the performance of *ThinkEdit*, highlighting that *ThinkEdit* is a more effective strategy addressing the accuracy drops of overly short reasoning.

Model		GSM8K	MMLU Elem. Math	MATH-Level1	MATH-Level5	MATH-500
deepseek-qwen-14B	ThinkEdit (8%)	94.30 ± 0.28	96.93 ± 0.50	96.09 ± 0.35	90.92 ± 0.41	91.26 ± 0.52
	ThinkEdit (4%)	93.78 ± 0.50	96.56 ± 0.84	96.36 ± 0.52	91.03 ± 0.44	91.92 ± 0.63
	ThinkEdit (2%)	93.50 ± 0.31	96.53 ± 0.54	96.50 ± 0.46	91.15 ± 0.59	91.78 ± 0.58
	Append "Wait"	91.30 ± 0.55	95.37 ± 0.70	96.52 ± 0.55	90.60 ± 0.41	91.22 ± 0.57
	Original	90.80 ± 0.36	95.08 ± 0.65	96.32 ± 0.35	90.25 ± 0.72	91.48 ± 0.55
deepseek-llama3-8B	ThinkEdit (8%)	90.18 ± 0.60	96.11 ± 0.67	94.39 ± 0.61	86.13 ± 0.46	87.64 ± 0.88
	ThinkEdit (4%)	89.44 ± 0.55	96.19 ± 0.73	94.44 ± 0.31	86.49 ± 0.54	88.06 ± 1.09
	ThinkEdit (2%)	88.97 ± 0.78	96.08 ± 0.86	94.12 ± 0.47	85.91 ± 0.48	87.60 ± 0.81
	Append "Wait"	84.28 ± 0.64	95.93 ± 0.70	93.96 ± 0.55	85.33 ± 0.79	87.66 ± 1.26
	Original	82.26 ± 0.91	96.01 ± 0.62	93.46 ± 0.84	85.49 ± 0.83	87.26 ± 1.16
deepseek-qwen-1.5B	ThinkEdit (8%)	84.81 ± 0.69	92.38 ± 1.04	94.00 ± 0.75	75.32 ± 1.11	82.56 ± 1.21
	ThinkEdit (4%)	84.56 ± 0.79	90.66 ± 0.97	93.66 ± 0.62	75.05 ± 0.82	82.24 ± 0.89
	ThinkEdit (2%)	83.34 ± 0.79	86.24 ± 1.12	93.89 ± 0.76	74.94 ± 0.85	82.74 ± 0.77
	Append "Wait"	79.81 ± 0.77	76.64 ± 1.18	93.34 ± 0.86	75.06 ± 0.72	82.98 ± 1.00
	Original	79.15 ± 1.08	68.52 ± 1.56	93.00 ± 0.33	75.48 ± 0.90	82.22 ± 1.29

Table 5: Overall accuracy (%) of ThinkEdit with different editing rates.

Model		GSM8K	MMLU Elem. Math	MATH-Level1	MATH-Level5	MATH-500
deepseek-qwen-14B	ThinkEdit (8%)	598.1 ± 1011.8	336.6 ± 550.3	1586.1 ± 1827.4	4150.5 ± 3819.1	3009.5 ± 3336.7
	ThinkEdit (4%)	538.2 ± 829.6	291.4 ± 607.5	1670.4 ± 1951.2	4243.7 ± 3814.0	3079.7 ± 3276.6
	ThinkEdit (2%)	479.8 ± 968.5	285.1 ± 756.8	1645.4 ± 1946.6	4327.2 ± 3863.4	3138.3 ± 3372.8
	Append "Wait"	447.3 ± 652.6	273.0 ± 215.8	1595.8 ± 1810.5	4265.9 ± 3749.0	3071.5 ± 3275.6
	Original	354.5 ± 684.4	184.9 ± 175.3	1600.5 ± 1885.2	4306.2 ± 3816.1	3096.8 ± 3308.0
deepseek-llama3-8B	ThinkEdit (8%)	971.8 ± 1447.7	1488.3 ± 1979.5	1692.8 ± 2200.5	4642.1 ± 4253.3	3463.3 ± 3800.1
	ThinkEdit (4%)	927.7 ± 1486.3	1517.9 ± 2041.5	1723.7 ± 2152.3	4773.5 ± 4327.4	3509.5 ± 3842.9
	ThinkEdit (2%)	849.7 ± 1454.8	1520.1 ± 2103.0	1765.7 ± 2315.1	4825.2 ± 4383.4	3503.8 ± 3838.4
	Append "Wait"	670.2 ± 1073.0	1514.4 ± 2009.1	1639.9 ± 2134.8	4795.3 ± 4296.2	3502.5 ± 3859.1
	Original	597.3 ± 1109.0	1486.6 ± 2036.7	1646.6 ± 2275.0	4789.1 ± 4315.4	3507.6 ± 3917.5
deepseek-qwen-1.5B	ThinkEdit (8%)	1166.2 ± 1986.4	890.7 ± 1851.7	1912.8 ± 2287.6	5567.4 ± 5083.4	3772.6 ± 4296.0
	ThinkEdit (4%)	1126.6 ± 2018.0	768.9 ± 1651.4	1946.3 ± 2438.4	5522.4 ± 5036.9	3821.1 ± 4384.9
	ThinkEdit (2%)	912.7 ± 1835.3	701.0 ± 1748.9	1918.0 ± 2420.6	5641.9 ± 5101.5	3880.3 ± 4402.4
	Append "Wait"	847.1 ± 1835.7	660.1 ± 1823.7	2163.7 ± 2847.0	6363.9 ± 5352.9	4287.1 ± 4710.3
	Original	768.1 ± 1837.2	517.0 ± 1502.8	2080.9 ± 2740.5	6360.0 ± 5336.4	4260.3 ± 4668.2

Table 6: Overall reasoning length (in tokens) of ThinkEdit with different editing rates.

Model		GSM8K	MMLU Elem. Math	MATH-Level1	MATH-Level5	MATH-500
deepseek-qwen-14B	ThinkEdit (8%)	96.46 / 97.02 / 97.38	97.22 / 95.95 / 95.73	98.57 / 97.91 / 97.47	98.48 / 98.56 / 98.22	91.60 / 93.00 / 94.60
	ThinkEdit (4%)	96.31 / 96.18 / 96.77	97.78 / 95.14 / 96.53	99.52 / 99.53 / 98.62	96.67 / 97.88 / 98.11	91.20 / 93.20 / 95.00
	ThinkEdit (2%)	96.62 / 96.03 / 96.12	96.11 / 96.22 / 96.27	100.00 / 99.77 / 98.85	95.76 / 97.65 / 98.07	89.60 / 92.60 / 94.70
	Append "Wait"	94.62 / 94.20 / 93.19	96.67 / 97.30 / 96.93	100.00 / 99.30 / 98.39	90.15 / 94.47 / 96.33	85.20 / 89.20 / 93.30
	Original	96.31 / 95.65 / 92.93	93.89 / 96.22 / 95.60	99.52 / 99.30 / 97.70	89.39 / 94.32 / 96.25	86.40 / 91.40 / 93.50
deepseek-llama3-8B	ThinkEdit (8%)	96.31 / 96.49 / 95.97	97.78 / 97.57 / 98.40	99.05 / 99.30 / 98.85	97.12 / 97.58 / 97.39	95.20 / 94.20 / 94.80
	ThinkEdit (4%)	96.31 / 95.50 / 94.68	97.78 / 97.57 / 97.60	99.05 / 99.07 / 97.82	95.76 / 97.42 / 97.46	95.60 / 93.80 / 95.40
	ThinkEdit (2%)	97.08 / 95.27 / 93.95	97.78 / 98.65 / 97.87	100.00 / 99.30 / 98.62	95.61 / 96.89 / 97.12	92.80 / 93.60 / 94.40
	Append "Wait"	88.15 / 89.01 / 88.29	97.78 / 97.57 / 97.87	98.57 / 97.21 / 95.75	79.55 / 89.02 / 93.45	86.40 / 86.00 / 90.70
	Original	88.92 / 87.18 / 85.82	97.22 / 96.49 / 96.80	97.14 / 94.88 / 94.83	78.64 / 88.79 / 93.41	82.00 / 81.40 / 88.30
deepseek-qwen-1.5B	ThinkEdit (8%)	95.38 / 94.20 / 92.97	93.89 / 92.70 / 91.87	94.76 / 96.05 / 96.90	96.21 / 97.20 / 96.78	94.00 / 93.60 / 94.40
	ThinkEdit (4%)	92.62 / 92.90 / 92.32	87.78 / 88.11 / 88.67	95.71 / 95.58 / 96.44	95.15 / 96.59 / 97.27	90.80 / 92.00 / 94.20
	ThinkEdit (2%)	92.46 / 92.37 / 92.05	77.22 / 80.54 / 79.73	96.19 / 95.81 / 97.36	93.79 / 95.83 / 95.80	92.80 / 94.40 / 94.90
	Append "Wait"	88.92 / 87.10 / 86.77	82.22 / 79.46 / 76.13	96.67 / 96.74 / 96.44	92.27 / 94.85 / 95.72	86.00 / 90.60 / 92.30
	Original	88.46 / 87.48 / 85.02	62.78 / 62.16 / 60.53	97.62 / 95.12 / 93.91	91.52 / 95.00 / 95.72	82.40 / 89.80 / 93.40

Table 7: Accuracy (%) on the top 5% / 10% / 20% shortest responses for ThinkEdit with different editing rates.

Model		GSM8K	MMLU Elem. Math	MATH-Lvl1	MATH-Lvl5	MATH-500
deepseek-qwen-14B	ThinkEdit (8%)	113.2 / 129.4 / 153.6	86.9 / 99.0 / 117.2	180.7 / 238.5 / 372.3	768.1 / 925.6 / 1136.0	414.7 / 573.9 / 759.0
	ThinkEdit (4%)	101.7 / 113.6 / 131.0	82.7 / 91.8 / 105.6	146.7 / 188.6 / 346.0	745.5 / 926.6 / 1163.7	361.3 / 559.3 / 764.6
	ThinkEdit (2%)	95.4 / 106.3 / 120.2	79.1 / 87.1 / 98.7	125.1 / 150.2 / 243.4	698.5 / 906.6 / 1157.2	270.2 / 492.6 / 733.3
	Wait	127.2 / 145.0 / 166.0	104.1 / 114.4 / 127.6	159.3 / 191.8 / 281.9	672.1 / 875.5 / 1132.1	293.6 / 495.7 / 720.6
	Original	76.6 / 86.5 / 99.1	65.8 / 72.2 / 80.6	93.7 / 114.3 / 188.6	628.8 / 858.4 / 1125.9	198.7 / 434.3 / 697.0
deepseek-llama3-8B	ThinkEdit (8%)	160.4 / 185.7 / 225.2	426.0 / 484.4 / 559.4	209.5 / 267.2 / 380.8	825.3 / 978.8 / 1190.7	422.6 / 567.4 / 759.5
	ThinkEdit (4%)	110.3 / 131.8 / 164.6	398.5 / 462.4 / 541.8	176.3 / 221.3 / 336.0	806.1 / 963.3 / 1185.1	372.5 / 553.2 / 772.9
	ThinkEdit (2%)	93.2 / 106.9 / 127.4	396.5 / 464.2 / 543.2	137.4 / 173.3 / 277.1	791.2 / 954.8 / 1185.1	305.2 / 506.3 / 737.6
	Wait	132.2 / 148.2 / 169.1	444.5 / 501.7 / 565.9	148.4 / 179.2 / 244.0	680.8 / 887.3 / 1147.1	277.9 / 452.1 / 693.5
	Original	73.0 / 83.1 / 96.6	371.0 / 438.1 / 518.2	80.3 / 97.2 / 130.3	617.9 / 854.9 / 1126.5	159.5 / 357.5 / 644.5
deepseek-qwen-1.5B	ThinkEdit (8%)	115.9 / 138.2 / 180.1	87.4 / 103.7 / 130.1	247.3 / 396.1 / 577.3	859.4 / 1003.7 / 1216.6	595.9 / 719.8 / 871.6
	ThinkEdit (4%)	103.3 / 118.9 / 144.8	80.6 / 92.5 / 112.9	172.7 / 336.9 / 543.6	853.0 / 1003.5 / 1221.9	530.8 / 676.0 / 837.4
	ThinkEdit (2%)	97.2 / 109.4 / 126.3	75.9 / 85.0 / 99.5	127.9 / 174.1 / 416.4	818.0 / 984.5 / 1214.3	435.0 / 612.9 / 800.6
	Wait	120.6 / 137.0 / 158.0	101.6 / 112.9 / 128.0	147.9 / 180.1 / 310.2	822.7 / 1020.9 / 1306.0	341.8 / 556.6 / 791.8
	Original	78.8 / 89.4 / 103.0	61.6 / 68.5 / 77.6	88.8 / 110.3 / 219.7	804.6 / 1017.9 / 1314.0	249.7 / 506.5 / 760.7

Table 8: Average reasoning length (in tokens) of the top 5% / 10% / 20% shortest responses for ThinkEdit with different editing rates.

A.6 ThinkEdit Results for 32B Reasoning Model

We report results for the larger deepseek-distill-qwen-32B model. Although ThinkEdit does not yield overall accuracy gains on the MATH-series datasets (Table 9), it consistently achieves higher accuracy on short reasoning responses similar to the smaller models (Table 11).

Model		GSM8K	MMLU Elem. Math	MATH-Level1	MATH-Level5	MATH-500
deepseek-qwen-32B	ThinkEdit (8%)	95.34 ± 0.23	98.10 ± 0.17	95.31 ± 0.38	91.16 ± 0.45	91.44 ± 0.57
	ThinkEdit (4%)	95.25 ± 0.25	98.02 ± 0.31	96.02 ± 0.42	91.31 ± 0.50	91.60 ± 0.65
	ThinkEdit (2%)	94.90 ± 0.34	97.49 ± 0.49	96.27 ± 0.54	91.31 ± 0.29	91.62 ± 0.74
	Append "Wait"	92.72 ± 0.54	96.16 ± 0.45	96.27 ± 0.39	91.32 ± 0.46	91.46 ± 0.51
	Original	92.97 ± 0.39	95.93 ± 0.83	96.41 ± 0.45	91.27 ± 0.53	91.62 ± 0.58

Table 9: Overall accuracy (%) of deepseek-distill-qwen-32B with different ThinkEdit edit-rates.

Model		GSM8K	MMLU Elem. Math	MATH-Level1	MATH-Level5	MATH-500
deepseek-qwen-32B	ThinkEdit (8%)	665.6 ± 762.8	312.3 ± 332.0	1548.6 ± 1473.4	3676.7 ± 3388.7	2665.6 ± 2885.1
	ThinkEdit (4%)	445.8 ± 684.7	287.7 ± 600.0	1484.7 ± 1587.7	3821.1 ± 3518.3	2736.4 ± 2948.8
	ThinkEdit (2%)	405.3 ± 620.5	238.8 ± 315.9	1485.3 ± 1622.1	3947.0 ± 3564.7	2816.1 ± 3029.2
	Append "Wait"	405.5 ± 519.0	280.6 ± 401.5	1484.8 ± 1619.1	4103.9 ± 3606.0	2878.8 ± 3029.3
	Original	306.2 ± 515.4	168.9 ± 105.3	1457.6 ± 1621.0	4100.7 ± 3608.7	2872.0 ± 3034.8

Table 10: Overall reasoning length (in tokens) for deepseek-distill-qwen-32B.

Model		GSM8K	MMLU Elem. Math	MATH-Level1	MATH-Level5	MATH-500
deepseek-qwen-32B	ThinkEdit (8%)	99.08 / 98.47 / 97.95	98.33 / 97.57 / 97.07	99.52 / 98.60 / 97.36	99.55 / 99.39 / 98.64	94.40 / 95.40 / 96.10
	ThinkEdit (4%)	98.92 / 97.71 / 97.83	97.78 / 97.57 / 97.20	100.00 / 100.00 / 98.74	98.03 / 98.64 / 97.99	92.00 / 94.40 / 95.80
	ThinkEdit (2%)	98.92 / 98.24 / 97.68	96.67 / 97.03 / 96.80	99.05 / 98.84 / 98.51	97.58 / 98.26 / 98.22	90.00 / 92.60 / 94.70
	Append "Wait"	97.08 / 96.03 / 95.21	95.00 / 96.76 / 96.27	99.52 / 99.30 / 98.05	94.09 / 96.89 / 97.61	84.80 / 90.40 / 93.20
	Original	98.31 / 97.18 / 96.20	97.78 / 97.03 / 95.87	100.00 / 100.00 / 98.97	93.03 / 96.36 / 97.35	86.40 / 92.00 / 94.00

Table 11: Accuracy (%) on the top 5% / 10% / 20% shortest responses for deepseek-distill-qwen-32B.

Model		GSM8K	MMLU Elem. Math	MATH-Lvl1	MATH-Lvl5	MATH-500
deepseek-qwen-32B	ThinkEdit (8%)	105.2 / 121.8 / 148.6	89.2 / 100.5 / 117.7	367.8 / 492.8 / 625.4	793.5 / 919.5 / 1094.6	567.1 / 677.0 / 811.1
	ThinkEdit (4%)	95.2 / 105.8 / 120.1	85.9 / 96.1 / 110.6	146.9 / 202.2 / 360.9	751.1 / 905.4 / 1101.0	446.7 / 600.0 / 768.9
	ThinkEdit (2%)	93.2 / 103.6 / 116.6	79.1 / 88.6 / 101.5	124.3 / 155.3 / 307.6	746.4 / 910.8 / 1123.7	371.3 / 563.0 / 759.8
	Append "Wait"	125.7 / 143.0 / 163.7	109.6 / 121.1 / 135.9	151.4 / 182.0 / 247.2	725.7 / 914.4 / 1153.4	328.4 / 521.3 / 739.4
	Original	76.7 / 86.7 / 99.6	69.3 / 76.1 / 84.3	89.9 / 109.4 / 149.6	672.7 / 886.7 / 1139.2	216.4 / 454.9 / 705.9

Table 12: Average reasoning length (tokens) of the top 5% / 10% / 20% shortest responses for deepseek-distill-qwen-32B.

A.7 Examples of Steering the Reasoning Length

To demonstrate the effect of steering the reasoning length, we present two examples from gsm8k in Figures 7 and 8. Each figure contains three different reasoning outputs:

- **Left:** The model’s response when steered towards shorter reasoning with $\alpha = -0.04$.
- **Middle:** The original unaltered response.
- **Right:** The model’s response when steered towards longer reasoning with $\alpha = 0.04$.

These examples highlight that steering along the reasoning direction effectively modulates the reasoning length without causing unintended truncation or artificial elongation. In contrast to naive methods such as forcibly stopping the reasoning process—resulting in incomplete reasoning—or appending redundant tokens like "Wait" to artificially extend responses, our approach enables the model to generate complete and coherent reasoning of varying lengths. By directly manipulating the internal representation, we achieve a more controlled and flexible mechanism for managing reasoning length, which could be particularly useful for constrained computational budgets.

Toulouse has twice as many sheep as Charleston. Charleston has 4 times as many sheep as Seattle. How many sheep do Toulouse, Charleston, and Seattle have together if Seattle has 20 sheep?

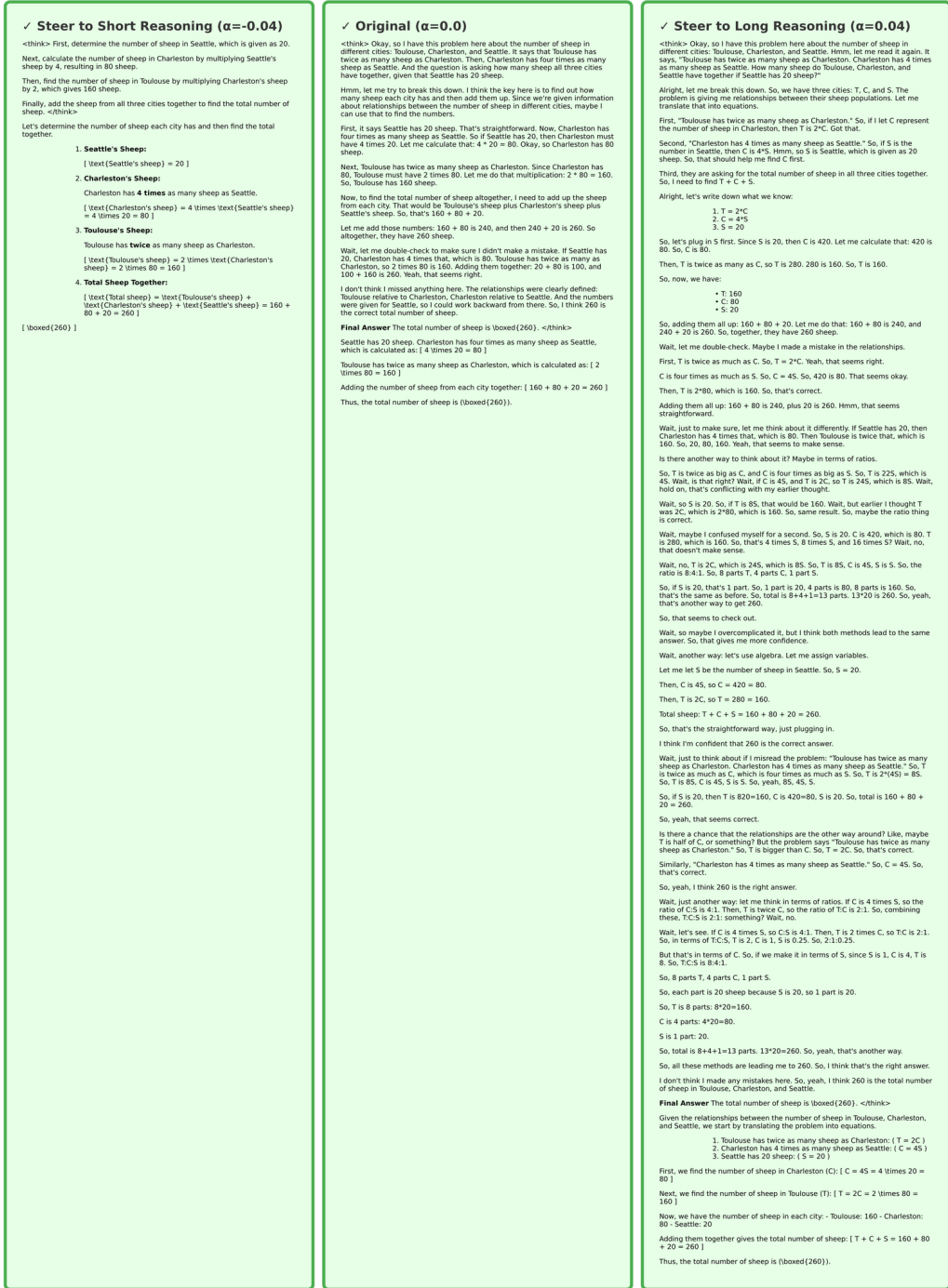


Figure 7: GSM8k Example 1: Steering the reasoning length of deepseek-distill-llama3-8b. Left: shorter reasoning ($\alpha = -0.04$), Middle: original response, Right: longer reasoning ($\alpha = 0.04$).

In a dance class of 20 students, 20% enrolled in contemporary dance, 25% of the remaining enrolled in jazz dance, and the rest enrolled in hip-hop dance. What percentage of the entire students enrolled in hip-hop dance?

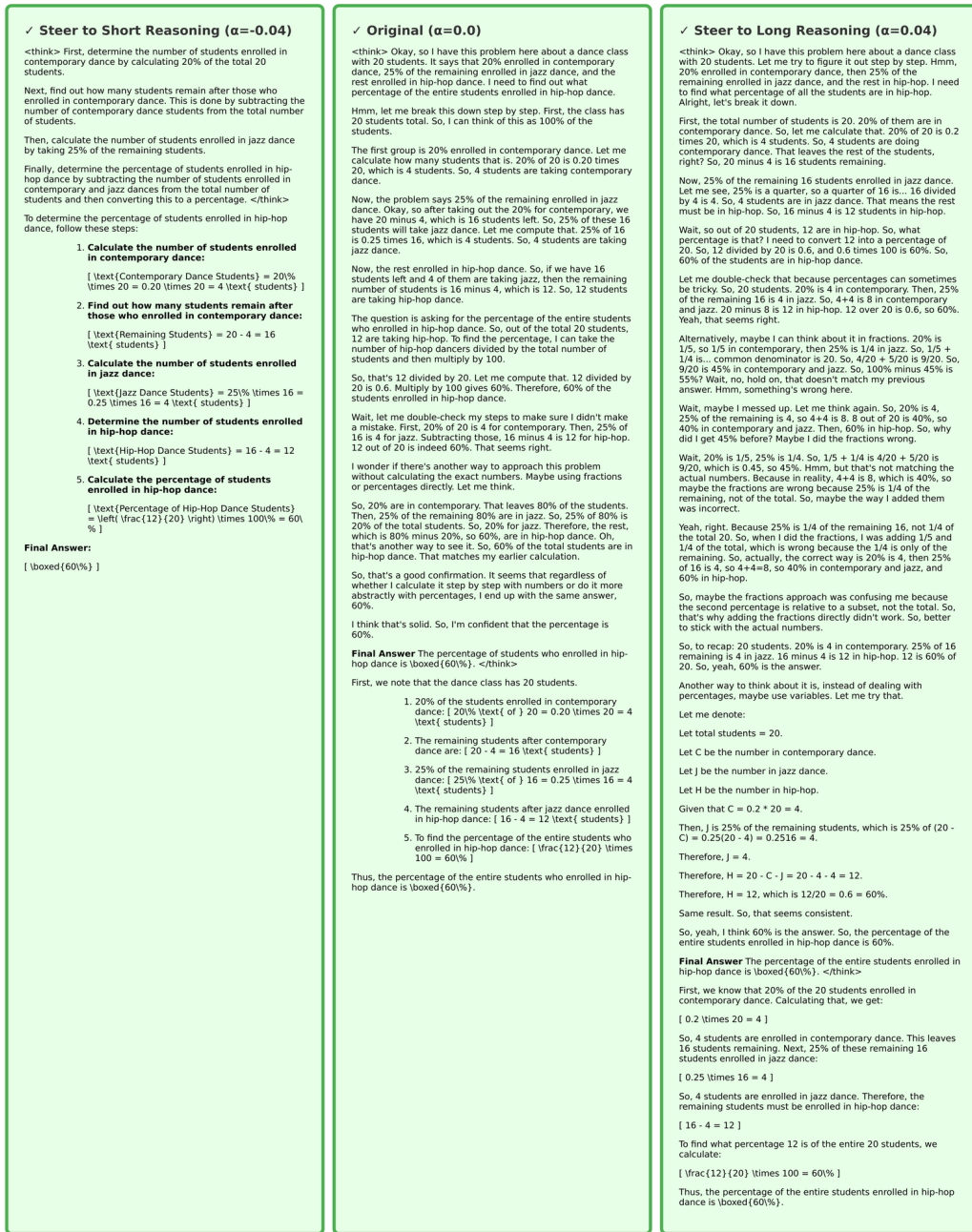


Figure 8: GSM8k Example 2: Steering the reasoning length of deepseek-distill-llama3-8b. Left: shorter reasoning ($\alpha = -0.04$), Middle: original response, Right: longer reasoning ($\alpha = 0.04$).

A.8 Examples of Fixing the Overly Short Reasoning with *ThinkEdit*

To illustrate the effectiveness of our *ThinkEdit* approach in addressing the overly short reasoning issue, we show two representative examples from the GSM8K and MATH-level5 datasets.

Example from GSM8K. The example is shown in Figure 9. In this problem, the *Original* model misinterprets the question and quickly provides an incorrect conclusion. In contrast, after applying *ThinkEdit*, the model first organizes the given information and displays a detailed reasoning process that leads to the correct answer.

Example from MATH-LEVEL5. The example is shown in Figure 10. Both the *Original* model and the model with *ThinkEdit* begin with an incorrect line of reasoning. However, the original model does not revise its approach and proceeds to an incorrect final conclusion. In contrast, the model with *ThinkEdit* revisits its reasoning steps, recognizes the mistake, and corrects the process in time to reach the correct solution.

These examples demonstrate *ThinkEdit* can facilitate deeper, more accurate reasoning by guiding the model to properly utilize problem details, reconsider faulty steps, and ultimately yield correct solutions.

The Doubtfire sisters are driving home with 7 kittens adopted from the local animal shelter when their mother calls to inform them that their two house cats have just had kittens. She says that Patchy, the first cat, has had thrice the number of adopted kittens, while Trixie, the other cat, has had 12. How many kittens does the Doubtfire family now have?

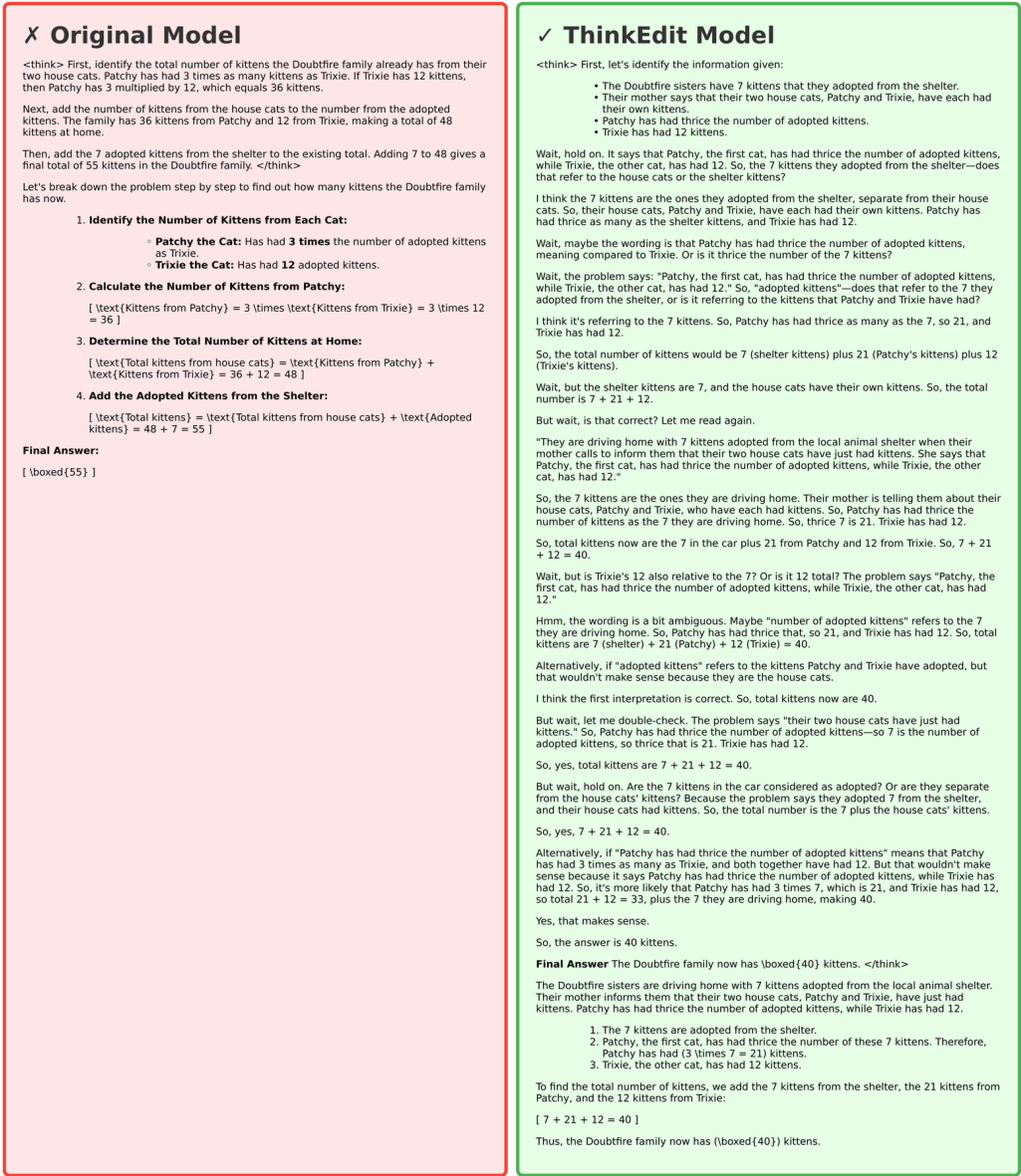


Figure 9: An example of *ThinkEdit* from the GSM8K dataset. The original model provides a short, flawed explanation. After *ThinkEdit*, the model’s reasoning process is more thorough.

At the national curling championships, there are three teams of four players each. After the championships are over, the very courteous participants each shake hands three times with every member of the opposing teams, and once with each member of their own team.

How many handshakes are there in total?

✗ Original Model

<think> First, determine the total number of participants. There are three teams, each with four players, making a total of 12 participants.

Next, calculate the number of handshakes within each team. Each team member shakes hands once with their teammates. For one team, this results in 4 players shaking hands with 3 others each, totaling 6 handshakes. Since there are three teams, the total intra-team handshakes are 18.

Then, calculate the number of handshakes between teams. Each player shakes hands three times with each member of the opposing teams. For one player, this involves 3 players from each of the other two teams, resulting in 6 handshakes per player. With 12 players in total, this gives 72 handshakes across all inter-team interactions.

Finally, add the intra-team and inter-team handshakes together to find the total number of handshakes: $18 + 72 = 90$.

Let's break down the problem step by step.

Total Participants: - There are 3 teams, each with 4 players. - Total participants = $3 \times 4 = 12$ players.

Handshakes Within Each Team: - Each player shakes hands **once** with each of their teammates. - For one team: - Number of handshakes = $(\binom{4}{2}) = \frac{4 \times 3}{2} = 6$. - Since there are 3 teams: - Total intra-team handshakes = $6 \times 3 = 18$ handshakes

Handshakes Between Teams: - Each player shakes hands **three times** with each member of the opposing teams. - For one player: - Number of opponents from other teams = 8 (since there are 12 - 4 = 8 players not in their team) - Total handshakes per player = $8 \times 3 = 24$ handshakes - For all 12 players: - Total inter-team handshakes = $12 \times 24 = 288$ handshakes

Total Handshakes: - Add intra-team and inter-team handshakes: - Total handshakes = $18 + 288 = 306$ handshakes

[boxed(306)]

✓ ThinkEdit Model

<think> To determine the total number of handshakes, I need to consider how often each participant shakes hands with others. Each person shakes hands three times with every opponent and once with their own team member.

First, let's think about the interactions within the same team. Each team has four players, and each one shakes hands once with the others. So for one team, that's 4 players each shaking hands once with 3 teammates, which is $4 \times 3 = 12$ handshakes, but since each handshake is between two people, I need to divide by 2 to avoid double-counting. So within one team, it's 6 handshakes.

But wait, actually, no. If each player shakes hands once with each team member, then for a team of four, each handshake is unique. So it's actually 4 choose 2, which is 6 handshakes. So yeah, 6 handshakes per team.

But there are three teams, so that's $3 \times 6 = 18$ handshakes within the same teams.

Now, between teams, each person shakes hands three times with every member of the opposing teams. So let's figure out how many inter-team handshakes there are.

Each team has four players, and there are three teams, so each player has $4 \times 3 = 12$ opponents. Each handshake is three times, so that's $12 \times 3 = 36$ handshakes per player. But since each handshake involves two people, we need to divide by 2 to avoid double-counting. So $36 \times 3 / 2 = 54$ handshakes between teams.

Wait, hold on. Let me think again.

If each of the 4 players on one team shakes hands three times with each of the 4 players on another team, that's $4 \times 4 \times 3 = 48$ handshakes between two teams. But since there are three teams, each pair of teams has 48 handshakes. How many pairs of teams are there? It's 3 choose 2, which is 3. So total inter-team handshakes would be $48 \times 3 = 144$.

But that seems high. Maybe I'm overcounting. Let me verify.

Alternatively, think of it as each pair of opposing players shakes hands three times. There are 3 teams, so each player is on one team, and there are two other teams. Each player has 4 opponents on each of the other two teams, so 8 opponents in total. Each of these 8 pairs shakes hands three times, so per player, that's $8 \times 3 = 24$ handshakes. Since there are 12 players in total (4 per team $\times$ 3 teams), that's $12 \times 24 / 2 = 144$ handshakes. So that matches.

So inter-team handshakes are 144.

Now, adding the intra-team handshakes: 6 per team $\times$ 3 teams = 18.

Total handshakes: $144 + 18 = 162$.

Wait, but that seems a bit high. Let me check another way.

Total number of participants is 12. Each person shakes hands with 11 others. But the number of handshakes isn't 12×11 because each handshake is counted twice. So total handshakes would be $(12 \times 11) / 2 = 66$, but that's if each pair shakes hands once. However, in this case, some pairs shake hands three times and some once.

So perhaps I need to calculate the total number of handshakes as follows:

Within the same team: Each team has 4 players, each shaking hands once with each of the other 3, so $4 \times 3 / 2 = 6$ handshakes per team. Three teams, so 18.

Between teams: For each pair of teams, each player on one team shakes hands three times with each player on the other team. So for two teams, that's $4 \times 4 \times 3 = 48$ handshakes. There are 3 pairs of teams, so $48 \times 3 = 144$.

Total handshakes: $18 + 144 = 162$.

Alternatively, let's think about each player's total handshakes. Each player shakes hands once with their 3 teammates and three times with each of the 8 opponents (since there are 4 opponents on each of the other two teams). So that's $3 + (4 \times 3 \times 2) = 3 + 24 = 27$ handshakes per player. Wait, 4 opponents on each of two teams, so $4 \times 2 = 8$ opponents, each with 3 handshakes, so $8 \times 3 = 24$, plus 3 teammates each with 1 handshake: $24 + 3 = 27$.

So each player has 27 handshakes, and there are 12 players, so total handshakes counted this way would be $12 \times 27 = 324$. But since each handshake is counted twice, we divide by 2: $324 / 2 = 162$. So that confirms it.

Therefore, the total number of handshakes is 162.

Final Answer The total number of handshakes is boxed(162). </think>

To determine the total number of handshakes at the national curling championships, we need to consider both intra-team and inter-team handshakes.

- Intra-team handshakes:**
 - Each team has 4 players, and each player shakes hands once with each of the other 3 teammates.
 - For one team, the number of handshakes is calculated as $(\binom{4}{2}) = 6$.
 - There are 3 teams, so the total number of intra-team handshakes is $(3 \times 6 = 18)$.
- Inter-team handshakes:**
 - Each player shakes hands three times with each member of the opposing teams.
 - Each player has 4 opponents on each of the other two teams, resulting in 8 opponents in total.
 - Each pair of opposing players shakes hands three times, so for each player, the number of handshakes is $(8 \times 3 = 24)$.
 - Since there are 12 players in total, the total number of handshakes counted this way is $(12 \times 24 = 288)$.
 - However, each handshake is counted twice, so we divide by 2: $(288 / 2 = 144)$.
- Total handshakes:**
 - Adding the intra-team and inter-team handshakes gives $(18 + 144 = 162)$.

Thus, the total number of handshakes is boxed(162).

Figure 10: An example of *ThinkEdit* from MATH-level5. While both models initially make a wrong assumption, the model after applying *ThinkEdit* corrects itself and arrives at the correct final reasoning.