TOWARDS REASONING REUSE: A NEW PARADIGM IN MODEL COLLABORATION

Zhengxi Li[1*] **Fuyuan Lyu**[2*] **Qiyuan Zhang**[3] **Ye Yuan**[2] **Haolun Wu**[2] **Xue Liu**[1,2]
[1]MBZUAI [2]McGill University & Mila [3]City University of Hong Kong

ABSTRACT

While large language models (LLMs) have demonstrated remarkable capabilities through training-time scaling and test-time scaling, increased costs constrain their deployment in applications. Existing works have developed fine-grained collaboration frameworks with small language models (SLMs). Despite their success in balancing performance and cost, these frameworks are hard to deploy broadly, since they are typically trained for a specific set of models and assume white-box access to all collaborating models. We propose **reasoning reuse**, a training-free model collaboration framework via test-time updates: an LLM first generates a limited number of reasoning steps, and an SLM reuses these steps to continue inference. This setting includes a large design space over what the LLM emits and how the SLM reuses it. In this paper, we establish feasibility: our experiments show that an SLM has the ability to reuse an LLM's reasoning steps. The ideas and findings in this work serve as an alternative framework for efficient language model collaboration at test time, paving the way for future work in this direction.

1 INTROUDCTION

Large language models (LLMs) have demonstrated remarkable capabilities through training-time scaling, which increases model size and data (Kaplan et al., 2020; Hoffmann et al., 2022), and test-time scaling, which allocates more computation during inference (Brown et al., 2024; Zhang et al., 2025). However, training-time and test-time scaling come with increasing computational costs that primarily constrain the use of LLMs for applications, especially advanced proprietary models (Singh et al., 2026; Anthropic, 2025; Comanici et al., 2025). To mitigate this issue, collaboration with small language models (SLM) becomes a practical compromise (Chen et al., 2024).

Prior approaches have proposed query-level routing and cascading to select the most suitable language model for different input queries (Chuang et al., 2025; Chen et al., 2024). These paradigms achieve great success in balancing performance and cost empirically (Hu et al., 2024) and theoretically (Dekoninck et al., 2025). Furthermore, recent works have proposed fine-grained token-level routers that enable model collaboration within a single query: an LLM only needs to produce critical tokens, since the remaining tokens can be routed to an SLM when it can contribute (Zheng et al., 2025; Fu et al., 2025). Despite the success of token-level routers in reducing costs, they lack portability and scalability in real-world applications. First, token-level routers are usually trained given the model sets, limiting their generalizability and transferability in deployment. In addition, token-level routers typically require all participating models to be white-box, preventing the collaboration with large, powerful, proprietary models.

To enable more generalizable, fine-grained model collaboration, we propose *reasoning reuse* (Figure 1),a training-free model collaboration framework that is compatible with proprietary models. In this framework, a strong LLM first generates a limited number of reasoning steps, and an SLM continues inference by taking these reasoning steps as part of the input prompt and reusing them as additional hints. We treat the number of an LLM's reasoning steps as budgeted test-time update signals: they do not change model parameters, but they update the inference context as guidance for subsequent generation. Regarding how to effectively use these test-time update signals, we raise two questions:

[*]Equal contribution

1. What type of reasoning steps should LLM provide such that SLM can effectively reuse them? Specifically, do the reasoning steps have to be partial solutions to a problem, or are there alternative forms, such as decomposing the problem into subproblems?
2. In what way should an SLM reuse an LLM's responses to fully leverage the information they contain as guidance for future generation? For example, what type of prompt helps the SLM understand how the LLM solves problems?

Reasoning reuse induces a combinatorial design space over what the LLM generates and how the SLM reuses it. Rather than claiming an optimal design in this space, we take a necessary first step by investigating a prerequisite question: **Can an SLM reuse an LLM's reasoning steps?** To answer it, we define the SLM's accuracy and token usage as two reasoning reuse indicators in § 2.2. Then, we conduct experiments to evaluate them when providing varying amounts of an LLM's reasoning steps. Furthermore, to study whether the effects of reasoning reuse come from the inherent capability of the SLM or by coincidence, we craft four prompting mechanisms that generally cover different levels of constraint imposed on the SLM. We define that an SLM possesses this ability if it can successfully reuse an LLM's steps under different prompts.

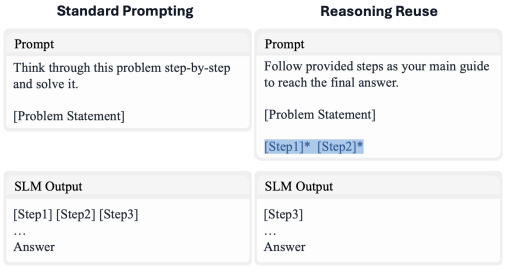

Figure 1: Reasoning Reuse: SLM takes LLM's reasoning steps as a test-time update signal and continues inference. LLM's reasoning steps are marked with [Step]* and highlighted.

2 PRELIMINARIES

2.1 REASONING REUSE

Two-stage reasoning. Let $q \in \mathcal{Q}$ denote a query (e.g., a math problem statement). The SLM $\mathcal{M}_S$ answers q with n reasoning steps $\mathbf{r}_{1:n}$:

$$\mathbf{r}_{1:n} \triangleq (r_1, \ldots, r_n) \sim p_{\mathcal{M}_S}(\mathbf{r}_{1:n} \mid q;\, \pi_S)\,, \tag{1}$$

where π_S denotes the *test-time specification* for $\mathcal{M}_S$ (e.g., system/user prompts, decoding configuration, and any other fixed conditioning). This is equivalent to a *two stage reasoning*:

$$\begin{aligned} \mathbf{r}_{1:k} &\triangleq (r_1, \ldots, r_k) \sim p_{\mathcal{M}_S}(\mathbf{r}_{1:k} \mid q;\, \pi_S)\,, \\ \mathbf{r}_{k+1:n} &\triangleq (r_{k+1}, \ldots, r_n) \sim p_{\mathcal{M}_S}(\mathbf{r}_{k+1:n} \mid q, \mathbf{r}_{1:k};\, \pi_S)\,. \end{aligned} \tag{2}$$

Reasoning reuse. At the first stage, LLM $\mathcal{M}_L$ replaces SLM to generate k reasoning steps $\mathbf{r}^*_{1:k}$. At the second stage, SLM conditions on both q and $\mathbf{r}^*_{1:k}$, and generates the remaining steps $\tilde{\mathbf{r}}_{k+1:m}$:

$$\begin{aligned} \mathbf{r}^*_{1:k} &\triangleq (r^*_1, \ldots, r^*_k) \sim p_{\mathcal{M}_L}(\mathbf{r}^*_{1:k} \mid q;\, \pi_L)\,, \\ \tilde{\mathbf{r}}_{k+1:m} &\triangleq (\tilde{r}_{k+1}, \ldots, \tilde{r}_m) \sim p_{\mathcal{M}_S}(\tilde{\mathbf{r}}_{k+1:m} \mid q, \mathbf{r}^*_{1:k};\, \tilde{\pi}_S)\,, \end{aligned} \tag{3}$$

which makes explicit that $\mathbf{r}^*_{1:k}$ serves as an *test-time update signals* for $\mathcal{M}_S$. Note that we use m to denote the number of steps (m might equal to n), because the reasoning prefix changes. $\tilde{\pi}_S$ controls the specific reusing methods vis prompting, see § 3.2 for details.

2.2 DEFINING SUCCESSFUL REASONING REUSE

In this sub-section, we define metrics for determining whether SLM can reuse reasoning steps.

Accuracy as an evaluation metric. Notabally, $(q, \mathbf{r}^*_{1:k})$ together is a more faithful query than $(q, \mathbf{r}_{1:k})$, because $\mathbf{r}^*_{1:k}$ in general has higher quality than $\mathbf{r}_{1:k}$. In addition, $\left(q, \mathbf{r}^*_{1:k+1}\right)$ together is an easier query than $(q, \mathbf{r}^*_{1:k})$, because $\mathbf{r}^*_{1:k+1}$ has one more step. We define a successful reusing as (notation $\|$ refers to concatenation of two reasoning traces):

1. The answer derived from $\mathbf{r}^*_{1:k} \| \tilde{\mathbf{r}}_{k+1:m}$ should have higher accuracy than $\mathbf{r}_{1:n}$.
2. The answer derived from $\mathbf{r}^*_{1:k+1} \| \tilde{\mathbf{r}}_{k+2:m}$ should have higher accuracy than $\mathbf{r}^*_{1:k} \| \tilde{\mathbf{r}}_{k+1:m}$.

Token usage as an evaluation metric. On one hand, since $\mathbf{r}^*_{1:k+1}$ is more informative than $\mathbf{r}^*_{1:k}$, it is true that if SLM can reuse reasoning steps, the token usage of $\tilde{\mathbf{r}}_{k+2:m}$ should be less than $\tilde{\mathbf{r}}_{k+1:m}$. On the other hand, however, it is not required that the token usage in $\mathbf{r}^*_{1:k} \| \tilde{\mathbf{r}}_{k+1:m}$ is less in $\mathbf{r}_{1:n}$: LLM might adopt a different solution with SLM, which can be a more token-consuming one; in this case, SLM should not be expected to reduce overall token usage of $\mathbf{r}^*_{1:k} \| \tilde{\mathbf{r}}_{k+1:m}$ below $\mathbf{r}_{1:n}$.

3 Experiments

3.1 Experiment setup

Benchmarks. We conduct experiments on *mathematics* datasets, AIME2024 and AIME2025 (Google, 2025). This setup ensures that the effect of any given reasoning step can be quantified by our step-wise evaluation metrics, as in mathematics problems each step can directly contribute to deriving the final answer.

LLM reasoning steps. To evaluate the SLM's reasoning reuse ability in isolation, we filter **absolutely correct** solutions (we do not evaluate error-identification ability) from GPT-5.1 Thinking (OpenAI, 2026), which is sufficiently strong to reach 100% and 97% pass@1 accuracy on the AIME2024/2025 benchmarks. We split each complete solution into sentences at sentence-final periods, and each sentence is treated as **one step**. Note that sentences stating the final answer are discarded.

Models. We test Phi-4 (Abdin et al., 2024), Qwen2-7B-Instruct (Yang et al., 2024), and Qwen3-8B (Team, 2025), enabling their thinking mode to generate inherent CoT for better reasoning performance (rather than using few-shot CoT prompting (Wei et al., 2022)).

3.2 Prompt Types

To validate whether an SLM inherently possesses the ability to reuse reasoning steps, rather than merely producing reuse-like generations by coincidence, we crafted four test-time reasoning reuse specifications with different levels of constraint (see Figure 2). **Follow** and **Complete** are instruction prompts, under which the SLM has some freedom to answer questions in other ways. In contrast, under **Inject** and **Inject***, the SLM must reuse the reasoning steps. Detailed prompt templates are provided in Appendix A.

4 Results

Baseline. In each prompt, three models are given 0% LLM reasoning steps.

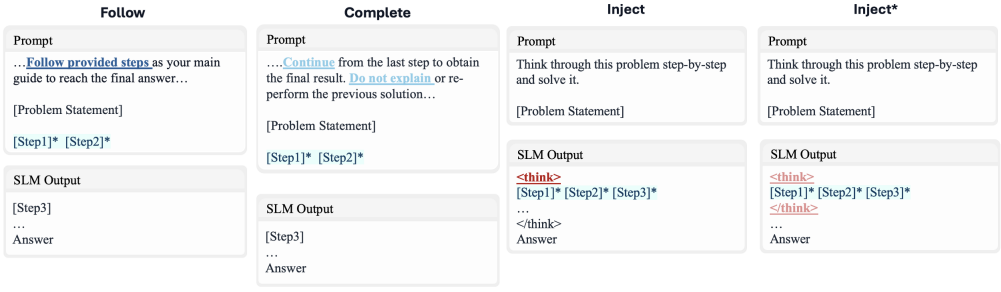

Figure 2: Fourt prompting mechanisms with different levels of constraint on SLM. The LLM's reasoning steps are marked with [Step]* and highlighted.

Rounding details. We introduce *step proportion* t to normalize the number of steps. The first k LLM's reasoning steps are mapped to a step proportion via $t(k) = k/N \times 100\%$, where N denotes the number of LLM reasoning steps. However, t is continuous whereas k is discrete; moreover, the total number of solution steps can vary across problems. As a result, the averaged accuracy and token usage at an arbitary step proportion t_1 are not well-defined. To address this, we estimate the averaged accuracy and token usage at t_1 using the results at the largest k such that $t(k) \leq t_1$.

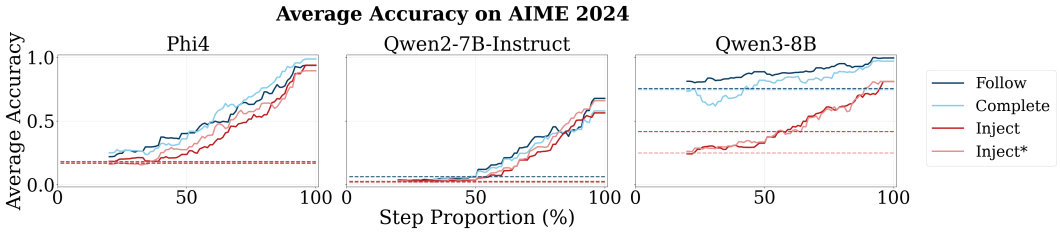

Figure 3: Accuracy of three SLMs in reusing GPT5.1-Thinking's reasoning steps across four prompting mechanisms. Baselines are shown as dashed lines.

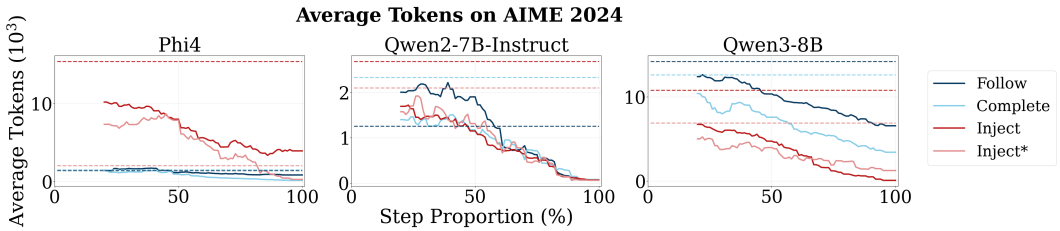

Figure 4: SLMs' token usage in reusing GPT5.1-Thinking's reasoning steps across four prompting mechanisms. Baselines are shown as dashed lines.

According to the metrics of successful reasoning reuse defined in § 2.2, we show that an SLM is able to reuse an LLM's reasoning steps. As shown in Figure 3, the results under each prompting mechanism surpass the baseline when given any step proportion (results on AIME2025 are in Appendix B.1). In addition, the average accuracy increases nearly monotonically as the LLM step proportion increases. For the token-usage metric, Figure 4 shows that the SLM's token usage nearly monotonically decreases when more reasoning traces are given. **These three pieces of evidence demonstrate that the SLM has the capability to reuse reasoning steps**.

In addition, the four prompts show similar accuracy-increasing trends on Phi4 and Qwen2-7B-Instruct. However, the **Inject** and **Inject*** accuracy scores are lower than those of the other two on Qwen3-8B. For the token-usage metric, the effects differ across the three models. We suggest that these phenomena might be caused by differences in the models' thinking mechanisms: Qwen3-8B explicitly supports thinking mode (Team, 2025), while the others do not (Abdin et al., 2024; Yang et al., 2024). Despite differences in absolute values across models and prompts, **similar trends indicate that the reasoning reuse capability comes from the SLM itself, not by coincidence.**

Interestingly, the three SLMs under particular reasoning reuse prompting mechanisms can consume both more and fewer tokens than the baseline. This phenomenon suggests that SLMs' preferred solutions in different settings might not be the same as the LLM's, and that the ways SLMs understand the LLM's solutions tend to be prompt- and model-specific.

5 Conclusion

In this work, we propose reasoning reuse as a test-time collaborative inference algorithm, aiming to balance performance and costs. We identify two key factors in reasoning reuse to offer insights for future work. Although we do not provide an optimal combination, we answer two principled questions using empirical evidence: our findings indicate that an SLM can reuse an LLM's reasoning steps, and that this uplifting effect comes from the model itself rather than by coincidence.

REFERENCES

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *ArXiv preprint*, abs/2412.08905, 2024.

AI Anthropic. System card: Claude opus 4 & claude sonnet 4. *Claude-4 Model Card*, 2025.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *ArXiv preprint*, abs/2407.21787, 2024.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. Featured Certification.

Yu-Neng Chuang, Prathusha Kameswara Sarma, Parikshit Gopalan, John Boccio, Sara Bolouki, Xia Hu, and Helen Zhou. Learning to route LLMs with confidence tokens. In *Forty-second International Conference on Machine Learning*, 2025.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *ArXiv preprint*, abs/2507.06261, 2025.

Jasper Dekoninck, Maximilian Baader, and Martin Vechev. A unified approach to routing and cascading for LLMs. In *Forty-second International Conference on Machine Learning*, 2025.

Tianyu Fu, Yi Ge, Yichen You, Enshu Liu, Zhihang Yuan, Guohao Dai, Shengen Yan, Huazhong Yang, and Yu Wang. R2r: Efficiently navigating divergent reasoning paths with small-large model token routing. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Google. Aime problems and solutions. `https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions`, 2025.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. An empirical analysis of compute-optimal large language model training. *Advances in neural information processing systems*, 35:30016–30030, 2022.

Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. Routerbench: A benchmark for multi-LLM routing system. In *Agentic Markets Workshop at ICML 2024*, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *ArXiv preprint*, abs/2001.08361, 2020.

OpenAI. Chatgpt (gpt-5.1 thinking). Large language model, February 2026. Model variant: gpt-5.1-thinking.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *ArXiv preprint*, abs/2601.03267, 2026.

Qwen Team. Qwen3 technical report, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *ArXiv preprint*, abs/2407.10671, 2024.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenyue Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, et al. A survey on test-time scaling in large language models: What, how, where, and how well? *ArXiv preprint*, abs/2503.24235, 2025.

Wenhao Zheng, Yixiao Chen, Weitong Zhang, Souvik Kundu, Yun Li, Zhengzhong Liu, Eric P. Xing, Hongyi Wang, and Huaxiu Yao. CITER: Collaborative inference for efficient large language model decoding with token-level routing. In *Second Conference on Language Modeling*, 2025.

A PROMPT

Follow Prompt

<|im_start|>system<|im_sep|>
You are a mathematical reasoner. You are given a math problem and a proposed solution with step-by-step reasoning. Follow and interpret the provided steps carefully to understand how the solution progresses. Use those steps as your main guide to reach the final answer. Put your final answer within \boxed{}.<im_end>

<im_start|>user<|im_sep|>
Problem:
{problem}

Proposed solution steps:
{steps} <im_end>

<im_start|>assistant<|im_sep|>

Complete Prompt

<|im_start|>system<|im_sep|>
You are a mathematical reasoner. You are given a math problem and a correct incomplete solution. Continue from the last step to obtain the final result. Do not explain or re-perform the previous solution. Derive the remaining steps to complete the solution and then give the final result. Output only what is important to show how you reached the final answer. Put your final answer within \boxed{}.<im_end>

<im_start|>user<|im_sep|>
Problem:
{problem}

Proposed solution steps:
{steps} <im_end>

<im_start|>assistant<|im_sep|>

Inject Prompt

<|im_start|>system<|im_sep|>
You are a careful and precise mathematical reasoner. Think through this problem step-by-step and solve it. Put your final answer within \boxed{}.<im_end>

<im_start|>user<|im_sep|>
Problem:
{problem} <im_end>

<im_start|>assistant<|im_sep|>
<think>
{steps}

Inject* Prompt

<|im_start|>system<|im_sep|>
You are a careful and precise mathematical reasoner. Think through this problem step-by-step and solve it. Put your final answer within \boxed{}.<im_end>

<im_start|>user<|im_sep|>
Problem:
{problem} <im_end>

<im_start|>assistant<|im_sep|>
<think>
{steps}
</think>

B Additional Results

B.1 AIME2025 Results

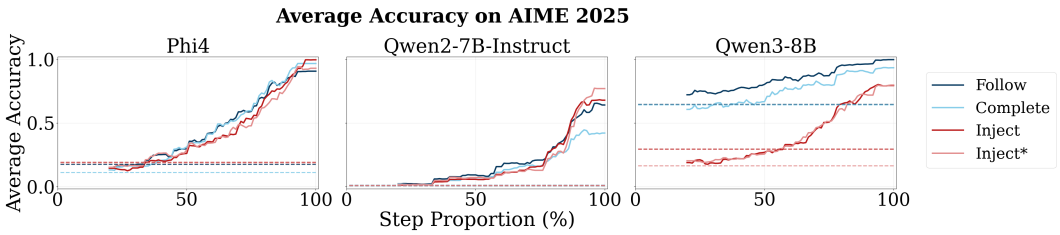

Figure 5: Accuracy of three SLMs in reusing GPT5.1-Thinking's reasoning steps across four prompting mechanisms. Baselines are shown as dashed lines.

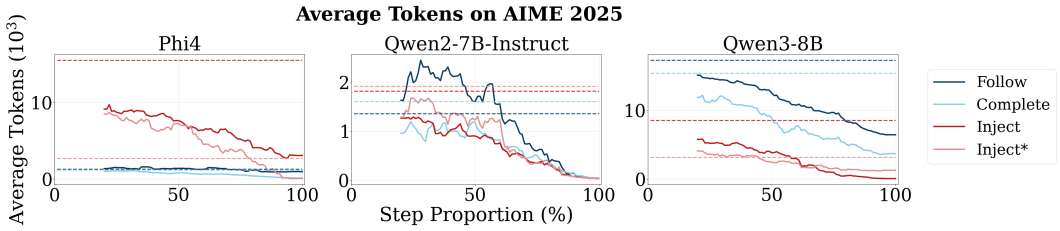

Figure 6: SLMs' token usage in reusing GPT5.1-Thinking's reasoning steps across four prompting mechanisms. Baselines are shown as dashed lines.

B.2 Number of Questions with Increased Accuracy

Table 1: Number of questions with increased accuracy on AIME2024 and AIME2025 datasets. We show that given 25%, 50%, 75%, and 100% reasoning steps, how many questions gain increased accuracy (maximum 30 on each dataset). For example, the number at top-left corner, 24, means that when Phi4 is given 25% reasoning steps, 24 questions gain increased accuracy on AIME2024.

Model	**Prompt**	**AIME2024**				**AIME2025**			
		25%	50%	75%	100%	25%	50%	75%	100%
Phi4	Follow	24	29	29	30	24	29	29	30
	Complete	26	26	30	30	26	29	30	30
	Inject	21	20	24	29	20	23	25	30
	Inject*	23	25	26	29	26	26	26	30
Qwen2-7b-Instruct	Follow	25	25	29	30	29	30	29	30
	Complete	28	29	30	30	27	28	29	30
	Inject	27	26	28	30	28	27	29	30
	Inject*	28	29	29	30	27	26	28	30
Qwen3-8b	Follow	25	27	28	29	24	27	27	30
	Complete	23	22	24	28	21	23	24	27
	Inject	8	14	22	23	13	19	22	26
	Inject*	22	24	26	28	27	28	29	29