

# The Challenge of Teaching Reasoning to LLMs Without RL or Distillation

Wei Du<sup>1</sup> Branislav Kisačanin<sup>1 2 3 4</sup> George Armstrong<sup>1</sup> Shubham Toshniwal<sup>1</sup> Ivan Moshkov<sup>1</sup>  
 Alexan Ayrapetyan<sup>1</sup> Sadegh Mahdavi<sup>1</sup> Dan Zhao<sup>1</sup> Shizhe Diao<sup>1</sup> Dragan Mašulović<sup>5</sup> Marius Stanean<sup>4</sup>  
 Advait Avadhanam<sup>6</sup> Max Wang<sup>7</sup> Ashmit Dutta<sup>8</sup> Shitij Govil<sup>9</sup> Sri Yanamandara<sup>8</sup> Mihir Tandon<sup>8</sup>  
 Sriram Ananthakrishnan<sup>10</sup> Vedant Rathi<sup>8</sup> David Zhang<sup>8</sup> Joonseok Kang<sup>8</sup> Leon Luo<sup>10</sup> Titu Andreescu<sup>4</sup>  
 Boris Ginsburg<sup>1</sup> Igor Gitman<sup>1</sup>

## Abstract

Reasoning-capable language models achieve strong performance in complex tasks by generating explicit and long Chain-of-Thought (CoT) traces. While prior work has shown that such reasoning can be learned through reinforcement learning or distillation from stronger models, we ask whether long CoT reasoning can be induced in base models with only prompting or minimal supervision. We demonstrate that as few as 20 high-quality CoT examples from the reasoning model QwQ-32B-Preview are sufficient to transform the base model Qwen2.5-32B into a reasoning model via fine-tuning. This result suggests that even extremely small-scale, high-quality reasoning supervision can unlock strong generalization capabilities. We further explore CoT supervision from non-reasoning models and human annotators, enhanced through prompt engineering, multi-pass editing, and structural guidance. However, these alternatives still fall short of the CoT traces from the reasoning model, suggesting that certain latent qualities in model-generated reasoning are not easily replicated. We also analyze key data attributes, such as problem difficulty, solution diversity, and output length, that contribute to effective reasoning transfer. To facilitate future research, we release our human-authored dataset across all refinement stages.

## 1. Introduction

Recent advances in large language models (LLMs) have led to remarkable progress in natural language understanding, code generation, and problem solving. Among the most notable developments is the emergence of reasoning models, such as OpenAI o1 (OpenAI, 2024), DeepSeek R1 (Guo et al., 2025), and Qwen QwQ (Team, 2025c), which demonstrate strong performance on complex tasks by producing long Chain-of-Thought (CoT) reasoning traces. Unlike traditional LLMs, which often produce direct or short CoT (Wei et al., 2022) responses, reasoning models can engage in exploratory reasoning processes. These traces typically include stages such as hypothesis generation, intermediate verification, and self-reflection behaviors that are widely regarded as key to their superior performance on complex tasks like advanced mathematics and competition-level programming.

Existing literature often associates reasoning models with the generation of long CoT, which are assumed to reflect strong problem-solving or “thinking” capabilities. However, early work (Wei et al., 2022) showed that short CoT prompting without any fine-tuning, can improve the reasoning abilities of LLMs. This observation motivates a central question: Can we induce reasoning mode characterized by reflective and self-exploration patterns in a non-reasoning base model using only a handful of high-quality long CoT examples? Specifically, we ask whether a small number of such examples (e.g., 10 to 50), provided via prompting or supervised fine-tuning (SFT), is sufficient to shift the model toward a reasoning mode characterized by stronger self-exploration and reflective problem solving.

To experimentally evaluate whether a base model is induced to reasoning mode, we first introduce a concrete and operational definition of reasoning behavior in the context of mathematical problem solving: A base model is said to exhibit reasoning behavior, or equivalently to operate in a reasoning mode, if after minimal supervision under long CoT reasoning patterns, it significantly outperforms a much larger non-reasoning model on challenging math

<sup>1</sup>Nvidia Corporation <sup>2</sup>Institute for Artificial Intelligence Research and Development of Serbia <sup>3</sup>Faculty of Technical Sciences, University of Novi Sad <sup>4</sup>AwesomeMath <sup>5</sup>Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad <sup>6</sup>Massachusetts Institute of Technology <sup>7</sup>Department of Computer and Information Science, School of Engineering and Applied Science, University of Pennsylvania <sup>8</sup>University of Illinois Urbana-Champaign <sup>9</sup>Georgia Institute of Technology <sup>10</sup>University of Chicago. Correspondence to: Wei Du <wedu@nvidia.com>.

The second AI for MATH Workshop at the 42<sup>nd</sup> International Conference on Machine Learning, Vancouver, Canada. Copyright 2025 by the author(s).

tasks. Since such a small number of examples is unlikely to convey substantial new factual knowledge, we interpret any observed performance gains not as a result of memorization or scale, but rather as evidence of a qualitative shift in problem-solving strategy, toward more iterative, exploratory, and reflective reasoning.

To test this hypothesis, we begin by evaluating the effectiveness of reasoning data. We use `QwQ-32B-Preview` (Team, 2025c) to generate high-quality CoT traces and fine-tune the base model `Qwen2.5-32B` (Yang et al., 2025). To evaluate the reasoning capabilities of the resulting checkpoints, we adopt the `Comp-Math-24-25 Benchmark` (Moshkov et al., 2025), a challenging dataset consisting of 256 mathematical problems requiring single numerical answers. These problems are drawn from the AIME and HMMT competitions held in 2024 and 2025. Surprisingly, we find that as few as 20 long CoT examples are sufficient to activate reasoning behavior in the base model. After fine-tuning on just these examples, the model achieves a 5.38% gain in **pass@1** and a 11.59% gain in **maj@64** over `Qwen2.5-Math-72B-Instruct` (Yang et al., 2024), one of the strongest open-source non-reasoning models.

Motivated by this result, we next explore whether similar reasoning behavior can be induced using solutions generated by non-reasoning models. Compared to human-written reasoning data, which are expensive and time-consuming to produce, non-reasoning data can be generated at scale with minimal effort. In this study, we select `Qwen2.5-32B-Instruct` (Yang et al., 2025) as a representative non-reasoning model, and explore various prompt formats and post-editing strategies using LLMs to generate long CoT solutions. Despite extensive efforts, we find it extremely challenging to generate solutions that exhibit genuinely reflective reasoning patterns. Even when using LLMs to post-edit non-reasoning outputs by explicitly inserting steps such as self-validation or reflection, the resulting traces often remain shallow and lack meaningful iterative reasoning. SFT on such data fails to activate reasoning behavior in the base model `Qwen2.5-32B`. Despite using orders of magnitude more data than in the distilled reasoning data, the resulting model only matches the performance of `Qwen2.5-Math-72B-Instruct`, and shows no signs of reasoning behavior.

Since non-reasoning data fail to induce reasoning behavior, we next investigate whether carefully designed human-written solutions can serve as an effective alternative. For this purpose we asked a group of volunteers, all of them former competitive math problem solvers, now college students, professors, and engineers, to solve a set of 50 problems. We documented four iterations of this effort - four versions of the dataset we call **Nemotron-Math-HumanReasoning** - as the solutions evolved over time.

We provide more details about this dataset and its versions in Sections 2.3 and 4.1 and open-source the data so other researchers can use this dataset as their starting point. Although we conducted several rounds of iterative refinement, SFT on human-written data still failed to shift the base model into reasoning mode.

To better understand what makes reasoning data effective, we performed a series of ablation studies focusing on various factors, including problem difficulty and diversity, solution length and correctness, and the presence of reasoning-related keywords. Interestingly, across all ablation settings, we observe that the reasoning data consistently succeed in activating reasoning behavior in the base model. We hypothesize that the underlying structure and demonstration patterns of the reasoning traces remain largely consistent across these variations, which may be the key factor responsible for triggering reasoning capabilities. Although we have found that as few as 20 long CoT reasoning examples with very limited SFT, can activate the model’s reasoning mode, we were unable to replicate this behavior using non-reasoning or human-written data, even after multiple rounds of refinement. Nevertheless, we remain optimistic that carefully curated human-written reasoning data, even in small quantities, could achieve similar effects. To support future research, we release our human-labeled dataset across multiple refinement stages and encourage the community to explore further what truly enables reasoning to emerge in LLMs.

## 2. Reasoning with Minimal Supervision

### 2.1. Training details

We perform SFT on `Qwen2.5-32B`, following a consistent training pipeline across all experiments. We conducted a grid search over various hyperparameter settings (Appendix D) and adopted the configuration that yielded the best performance. We employ the AdamW optimizer (Loshchilov and Hutter, 2019) with a fixed learning rate of  $1e-5$  and no warmup steps. Training is conducted with a batch size of 1,024 samples, and we adopt sequence packing and context parallelization techniques from NeMo-Aligner (Shen et al., 2024) to accelerate learning on long-context reasoning data. For datasets with fewer than 1,024 samples, the entire dataset is used as a batch. The context parallel size is set to 4, the tensor model parallel size is set to 8, and the packing length is fixed at 16,384 tokens. Each model is fine-tuned for 50 steps, and we use the final checkpoint as the output model. All of the experiments are done using NeMo-Skills<sup>1</sup>.

<sup>1</sup><https://github.com/NVIDIA/NeMo-Skills>

## 2.2. Evaluation Setup

To evaluate the performance of the reasoning models, it is essential to select a challenging dataset. We adopt the CompMath-24-25 Benchmark (Moshkov et al., 2025), which comprises 256 advanced-level mathematical problems that require single numerical answers, making evaluation both simpler and more accurate. As a non-reasoning baseline, we evaluate `Qwen2.5-Math-72B-Instruct`, one of the strongest open-source math models currently available. We generate solutions using 64 random seeds, with a temperature of 0.7, a top-p value of 0.95, and a maximum generation length of 16,384 tokens, even though, for this model, the longest correct solution observed on this benchmark is only 8,910 tokens. Evaluation is based on two metrics: **pass@1**, which measures the average accuracy across 64 independent runs, and **maj@64**, which reflects the accuracy of the majority vote among the 64 generations. The results for `Qwen2.5-Math-72B-Instruct` are shown in Figure 1 as the dashed blue line (**pass@1**) and the dashed orange line (**maj@64**). Throughout this paper, all models are evaluated using the same configuration.

## 2.3. Seed Problem Set for our Dataset

The **Nemotron-Math-HumanReasoning**<sup>2</sup> dataset begins with 50 randomly selected math problems sourced from the AoPS forums (of Problem Solving). These problems vary in difficulty but generally fall within the range of AIME to USAMO levels (of America). Most of them align with standard high school competition topics, such as number theory, combinatorics, algebra, and geometry, though a few also involve functions and calculus. All problems demand substantial reasoning and multi-step derivations to arrive at the final answer. Below is a representative example from the dataset:

*Find all integral solutions of the equation  $x^2 + 2y^2 = z^2$ .*

The final answer after the full derivation is:

$$(x, y, z) = (|2a^2 - b^2|k, 2abk, (2a^2 + b^2)k)$$

where  $b$  is odd,  $\gcd(a, b) = 1$ , and  $k \in \mathbb{Z}$ .

## 2.4. Inducing Reasoning with Minimal Supervision

It is worth exploring whether a base model can exhibit reasoning capabilities when exposed to only a small amount of high-quality supervision. To investigate this, we conducted experiments using reasoning data generated by `QwQ-32B-Preview`, varying the number of training examples from 10 to 50. Specifically, we used the same fixed set of 50 math problems as described in Section 2.3, and generated corresponding solutions using

`QwQ-32B-Preview`. For each problem, we generated 512 solutions using different random seeds and selected a single solution with the correct final answer. We thus obtained one long CoT reasoning solution per problem, resulting in 50 solutions in total. From this set, we randomly sampled 10, 20, 30, 40, and 50 examples to fine-tune the base model `Qwen2.5-32B`.

To ensure a fair comparison with human-written and non-reasoning data in subsequent experiments, we sampled or filtered the data to select solutions whose lengths are approximately close to 3K tokens. Although not exactly 3K, the average token lengths across all datasets with varying numbers of examples range from 3,400 to 3,800. This setup enables us to assess whether a small number of high-quality reasoning demonstrations can effectively elicit reasoning behavior in the base model. As illustrated in Figure 1 (solid blue line for **pass@1** and solid orange line for **maj@64**), the model notably outperforms the baseline `Qwen2.5-Math-72B-Instruct` with just 20 training examples, achieving a 5.38% absolute gain in **pass@1** (increasing from 11.72% to 17.10%) and an 11.59% improvement in **maj@64** (from 16.14% to 27.73%). This observation aligns with findings from (Muennighoff et al., 2025), which report that a limited amount of high-quality reasoning data, e.g., a dataset with just 1K examples, can substantially improve performance on mathematical and coding reasoning tasks. We also repeated the SFT experiments on `Qwen2.5-7B` and `Qwen2.5-14B` using the same data and settings. However, neither model is able to effectively enter reasoning mode, with SFT performance comparable to or worse than the baseline (see Appendix C). Therefore, we use `Qwen2.5-32B` for all experiments presented in this paper.

## 3. Generating Reasoning Data from Non-Reasoning Models

Motivated by this result, we first investigate whether similar reasoning behavior can be induced using solutions generated by non-reasoning models, as such data is easy to generate at scale and incurs minimal cost. We use `Qwen2.5-32B-Instruct` (Yang et al., 2025) as the non-reasoning model to collect long CoT generations. We selected `Qwen2.5-32B-Instruct` because it follows instructions more effectively than math-specific large language models such as `Qwen2.5-Math-72B-Instruct`.

### 3.1. Prompting Strategies for Generating Synthetic Reasoning Data

To better elicit long CoT outputs from the non-reasoning model `Qwen2.5-32B-Instruct`, we designed two prompting strategies (see Appendix A.1 and A.2): one based

<sup>2</sup><https://huggingface.co/datasets/nvidia/Nemotron-Math-HumanReasoning>

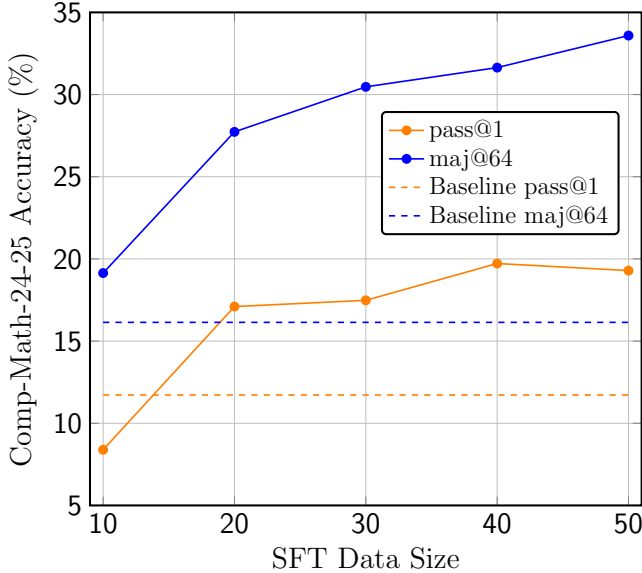


Figure 1. SFT accuracy of Qwen2.5-32B on Comp-Math-24-25 under varying data sizes. With only 20 examples, it outperforms the baseline Qwen2.5-Math-72B-Instruct. All datasets have average token lengths between 3,400 and 3,800.

solely on detailed instructional guidance, and another that augments the same instructions with the same 20 explicit reasoning examples as Section 2.4 used to conduct SFT on Qwen2.5-32B. Using the 50 problems described in Section 2.3, we generated 1,024 responses per problem using different random seeds. We retained only those outputs that produced the correct final answer and were between 1K and 8K tokens in length. This filtering yielded 2,499 valid samples from the instruction-only prompt (Appendix A.1) and 4,953 from the example-augmented prompt (Appendix A.2). According to the first two rows of Table 1, the example-augmented prompt generates a greater amount of data, with an average length of 1,214, slightly exceeding the 1,162 average from the instructional-only prompt.

We then fine-tuned the base model, Qwen2.5-32B, on each of these datasets. The performance of the resulting models on Comp-Math-24-25 is shown in the first two rows of Table 1. Although we used significantly more than 50 solutions, both prompts resulted in similar performance from the baseline Qwen2.5-Math-72B-Instruct. This suggests that prompting a non-reasoning model to generate sufficiently high-quality training data remains challenging and may be insufficient to transition the base model into a reasoning-capable regime. For comparison, the last two rows of Table 1 show the results of in-context learning using the same 20 examples as few-shot prompts for Qwen2.5-32B and Qwen2.5-32B-Instruct, which perform worse than the SFT counterparts.

### 3.2. Editing Synthetic Solutions to Induce Reasoning Patterns

The previous section demonstrates that prompting alone is insufficient to induce non-reasoning models to generate solutions exhibiting structured reasoning, such as self-verification, intermediate checks, or illustrative examples. Consequently, the resulting training data lack the necessary reasoning characteristics to transform the base model into reasoning model. This limitation raises a natural question: can LLMs be leveraged to revise existing non-reasoning solutions, enriching them with reasoning characteristics such as self-validation, self-check, or reflective examples?

To explore this, we used Qwen2.5-32B-Instruct to edit the original solutions via prompt-based instruction. We designed three prompts to encourage the insertion of reasoning behaviors: (i) one containing only detailed instructional guidance (Appendix B.1); (ii) one combining the same guidance with the 20 explicit reasoning examples used in Section 2.4 (Appendix B.2); and (iii) one instructing the model to incorporate an incorrect attempt into a revised reasoning-based solution (Appendix B.3). For the third prompt, we randomly selected an incorrect response from the same data pool and paired it with a correct solution, prompting the LLM to merge them into a single, improved response that reflects both reasoning and correction.

For data editing, we used the solutions generated by the example-augmented prompt (Appendix A.2). Rows 3 - 5 of Table 1 report the SFT performance on these revised solutions. Although the average length of the revised solutions increased compared to the original ones, indicating the insertion of some reasoning patterns, the resulting fine-tuned models showed no meaningful improvement. Their performance remained comparable to both the baseline model and the model fine-tuned on the original solutions. This suggests that the inserted reasoning patterns were superficial and insufficient to activate the base model’s reasoning capabilities.

## 4. Human-Crafted Solutions with Structured Reasoning

### 4.1. Progressive Refinement of Human-Written Solutions

We created four progressively refined versions of solutions for the same 50 problems, with each version incorporating increasingly detailed and structured reasoning:

1. In the first version, we asked our volunteers, all experienced competitive math problem solvers, to provide comprehensive solutions to each problem. We emphasized the importance of explicitly documenting their thought process in great detail and instructed them not



	Data	Pass@1	Maj@64	Size	Average Length
1	Non-reasoning data (Prompt A.1)	12.26%	14.84%	2,499	1,162
2	Non-reasoning data (Prompt A.2)	11.49%	14.45%	4,953	1,214
3	Non-reasoning data edit (Prompt B.1)	12.08%	15.62%	4,953	1,384
4	Non-reasoning data edit (Prompt B.2)	12.36%	15.75%	4,953	1,345
5	Non-reasoning data edit (Prompt B.3)	11.10%	15.23%	4,953	1,406
6	Human-written data ver. 1	5.51%	15.23%	50	2,679
7	Human-written data ver. 2	5.04%	13.28%	50	2,840
8	Human-written data ver. 3	3.88%	10.94%	50	3,088
9	Human-written data ver. 4	4.47%	13.28%	50	3,171
10	Human-written data ver. 4 edit (Prompt B.1)	7.82%	17.97%	50	2,591
11	Human-written data ver. 4 edit (Prompt B.2)	10.03%	15.62%	50	1,367
12	In-context prompting Qwen2.5-32B	5.38%	13.28%	N/A	N/A
13	In-context prompting Qwen2.5-32B-Instruct	9.62%	13.28%	N/A	N/A
14	Qwen2.5-32B on 50 examples	19.29%	33.59%	50	3,444
15	Qwen2.5-Math-72B-Instruct	11.72%	16.14%	N/A	N/A

Table 1. Accuracy on Comp-Math-24-25 using non-reasoning and human-written data. Rows 1–5 report the SFT accuracy on non-reasoning data and its edited versions. Rows 6–11 present the SFT accuracy on human-written data and its edited versions. Rows 12–13 show the accuracy of few-shot prompting using Qwen2.5-32B and Qwen2.5-32B-Instruct. Row 14 reports the SFT accuracy of Qwen2.5-32B on 50 examples from QwQ-32B-Preview, while Row 15 shows the baseline accuracy of Qwen2.5-Math-72B-Instruct.

to imitate or copy the style of reasoning models such as DeepSeek R1. As a result, the reasoning in these solutions was articulated with significant depth, often surpassing the level of explanation typically found in textbooks or problem collections.

2. After reflecting on how the initial solutions could be improved to better elicit reasoning, we adopted a more structured format with clearly marked phases. Each solution was broken down into steps, using section headers such as “Let’s restate the problem”, “Let’s consider a straightforward approach”, “But wait...” (to indicate a mistake or obstacle), “Let’s try another approach” and finally “I think this approach works!” followed by “Final answer”. These labels helped signal the reasoning process explicitly and encouraged a more transparent, exploratory style.
3. For the third version, we drew inspiration from a teaching strategy often used with human students, solving significantly simplified versions of the problem or attempting to guess simple cases. We refer to this technique as “going back to the first grade.” In our experience, this strategy helps students better understand the underlying structure of the problem, often more effectively than merely restating it. By attempting to solve special cases or simplified versions, students (and

models) can uncover key insights that are applicable to the general problem. To incorporate this idea, we added a new section immediately after “Let’s restate the problem,” beginning with: “Let’s take some special cases”.

4. As we examined why reasoning model outputs often elicit stronger reasoning behavior than human-written solutions, we observed a key style difference. Model-generated solutions tend to follow an exploratory process, revealing intermediate steps as they emerge. For example, a model might find that  $\angle A = \angle B$ , then  $\angle B = \angle C$ , and finally conclude that  $\triangle ABC$  is equilateral. In contrast, human solutions are usually composed after solving the problem through prior and informal reasoning. The final written solutions often reflect reconstructed reasoning, presenting conclusions before the supporting steps. This results in a non-causal narrative that masks the actual discovery process. To address this, we revised our solutions to enforce logical causality by presenting steps in the order they might be discovered. This restructuring aims to better mirror the reasoning style of models and more effectively prompt reasoning behavior in language models.

Rows 6 to 10 in Table 1 show the SFT accuracy for the four versions of human-written data. Notably, the average

length of these solutions is approximately 3K tokens, which is much longer than typical textbook solutions. However, even though we explicitly included reasoning steps and carefully documented human thought processes, the accuracy is still much lower than that of models trained on reasoning data. Even when compared to non-reasoning data, which lacks explicit reasoning, the human-written data achieves comparable **maj@64** scores but substantially lower **pass@1** scores.

We hypothesize that this is partly due to style variation in the human-written data. Since the solutions were authored by multiple individuals, there is considerable inconsistency in how reasoning is presented. In contrast, the non-reasoning and reasoning datasets generated by LLMs exhibit much more consistent and homogeneous styles, which likely help the model learn more stable reasoning patterns. The combination of limited data scale and high stylistic variance in the human-written data makes it more difficult for the model to generalize effectively. In contrast, the consistency in LLMs-generated data likely facilitates more effective learning, even under similar data size constraints.

## 4.2. LLMs-Guided Editing of Human-Written Solutions

We use the same first two prompts B.1 and B.2 from Section 3.2, excluding the third prompt since human-written data do not contain incorrect solutions, to revise the human-written data and encourage the insertion of reasoning behaviors.

Although we instructed the models to retain the original text and only insert reasoning behaviors, we still observed a decrease in the average length of the revised human-written solutions. We attribute this to the LLMs removing redundant or repetitive content, resulting in more concise and well-structured responses. As shown in Rows 10–11 of Table 1, the SFT accuracy on the edited human-annotated data shows a slight improvement, ultimately aligning with the non-reasoning baseline but not surpassing it. We believe the improvements come from LLM-assisted editing, which helps make the structures more consistent, though not enough to trigger reasoning behavior in the base model.

## 5. Which Factors Influence Reasoning Ability in Training Data?

### 5.1. The Role of Solution Correctness in Reasoning Learning

In the previous sections, we selected training data based on the correctness of the final answer, assuming that only complete and correct reasoning trajectories are beneficial for eliciting reasoning capabilities in base models. However, it remains unclear whether partially correct solutions that

follow the correct output format but arrive at an incorrect final answer can still offer learning value. To explore this, we selected 50 solutions from the same data pool in Section 2.4 that produced the correct final answer format but were ultimately incorrect in their final conclusions. Despite the errors, it is possible many of these solutions contained valid intermediate reasoning steps that aligned with correct problem-solving strategies.

The SFT results in Table 2 show that both the 50 correct (Row 8) and the 50 incorrect solutions (Row 1) were able to activate the base model’s reasoning ability to a comparable extent. This suggests that even solutions with incorrect final answers can provide valuable supervision signals, as they often contain structurally sound and logically coherent intermediate reasoning steps. Rather than simply memorizing knowledge from these examples, the model appears to benefit from exposure to the reasoning process itself, rather than what to think. These findings support the view that effective reasoning supervision does not necessarily require fully correct answers, as long as the underlying reasoning structure remains instructive.

### 5.2. Does the Presence of Keywords Matter in Reasoning Data?

Reasoning data often includes phrases such as “but wait”, “let me think” or “let me check” which are hypothesized to encourage behaviors like backtracking, self-verification, and reflection mechanisms believed to enhance a model’s reasoning ability. To assess the actual impact of these keywords, we took the same 50 solutions from QwQ-32B-Preview in Section 2.4, and removed the top 10 most frequent reasoning-related phrases (e.g., “but wait”, “maybe I can consider” etc.).

The second row of Table 2 shows that removing these keywords results in comparable performance, suggesting that the model’s reasoning ability is not primarily driven by the presence of such keywords. Instead, this supports the view that it is the underlying structure of high-quality reasoning demonstrations, rather than specific keywords, that activates the model’s reasoning capabilities. Notably, similar behavior has also been observed in (Li et al., 2025a).

### 5.3. Impact of Problem Difficulty on Model Reasoning

To investigate whether problem difficulty influences the utility of reasoning data, we used the pass rate of the 50 problems, calculated as the average success rate over 512 generations from the QwQ-32B-Preview model. Based on these scores, we categorized the problems into three difficulty levels: easy (pass ratio 0.7–1.0), medium (0.3–0.7), and hard (0.0–0.3). From each level, we randomly selected 10 problems and retained five correct solutions per problem, keeping the total dataset size consistent at 50. We matched

	Data	Pass@1	Maj@64	Average Length
1	Incorrect Solutions	21.21%	33.59%	3,303
2	Without keywords	18.70%	33.20%	3,279
3	Difficulty Level (0.0 - 0.3)	19.38%	35.94%	3,334
4	Difficulty Level (0.3 - 0.7)	20.95%	34.38%	3,822
5	Difficulty Level (0.7 - 1)	21.20%	31.64%	3,320
6	10 Problems	21.19%	33.20%	3,445
7	25 Problems	20.87%	33.59%	3,729
8	50 Problems	19.29%	33.59%	3,433
9	2K average length	19.67%	26.95%	2,421
10	4K average length	21.82%	33.98%	4,008
11	6K average length	23.30%	34.77%	6,049
12	8K average length	22.25%	37.11%	7,898

Table 2. Model accuracy on Comp-Math-24-25 under different reasoning dataset constructions, with each dataset containing 50 examples.

the average solution length across difficulty levels, targeting approximately 3K tokens per solution. This process yielded three reasoning datasets (easy, medium, and hard), each containing 50 correct solutions with comparable lengths.

We fine-tuned the base model Qwen2.5-32B on each of these datasets. Interestingly, the resulting model performances (Rows 3 - 5 in Table 2) were comparable across all difficulty levels, suggesting that problem difficulty alone does not significantly affect the model’s ability to acquire reasoning skills. These findings are consistent with our previous observation: the presence of explicit reasoning demonstrations in the data is more critical than the difficulty of the problems themselves.

#### 5.4. Impact of Problem Diversity on Model Reasoning

To investigate the impact of problem diversity on the effectiveness of reasoning data, we fixed the total number of training examples at 50 while varying the number of unique problems. We still use the 50 problems set in Section 2.3 and construct the dataset as follow: (1) 10 unique problems, each paired with 5 distinct solutions; (2) 25 unique problems, each with 2 distinct solutions; and (3) 50 unique problems, each with 1 solution. The 10 and 25 problem subsets were randomly sampled from the same pool of 50 problems. We reused the same generated solutions as in Section 2.4, filtering only for those with correct final answers.

Interestingly, we found that varying the number of unique problems, while keeping the total number of examples, had minimal impact on model performance (Rows 6 - 8 in Table 2). Models trained on highly diverse datasets (50 problems  $\times$  1 solution) performed comparably to those trained on more repetitive ones (10 problems  $\times$  5 solutions). These results

once again reinforce our earlier findings: it is the presence of high-quality reasoning demonstrations, rather than the diversity or difficulty of the problems, that most strongly drives reasoning behavior in models.

#### 5.5. The Influence of Solution Length on Reasoning Data Quality

As we know, the most notable characteristic of reasoning data is its substantial length, which enables it to explore multiple reasoning paths and perform self-validation. To investigate the impact of example length on SFT performance, we constructed four datasets with lengths of 2K, 4K, 6K, and 8K tokens, using the same data pool described in Section 2.4. Since each problem has 512 generated solutions, we are able to filter correct solutions based on their length. Each dataset contains the same 50 problems, with one correct solution per problem. The only variation across datasets lies in the length of these solutions. The results in Table 2 (Rows 9–12) show a slight performance improvement as the solution length increases. This improvement may be attributed to longer solutions exhibiting more frequent reasoning patterns and providing elaborate demonstrations of the reasoning process, thereby more effectively activating the model’s reasoning capabilities, especially given the limited dataset size.

### 6. Related Work

#### 6.1. Long CoT Reasoning

Chain-of-Thought (CoT) prompting has been empirically shown to be an effective method for enhancing the reasoning capabilities of LLMs by encouraging the generation of

intermediate logical steps. Initially introduced by Wei et al. (Wei et al., 2022), this approach demonstrated strong performance across a wide range of reasoning tasks, including arithmetic computation, commonsense inference, and symbolic manipulation. While numerous follow-up studies ((Zhang et al., 2022; Wang and Zhou, 2024; Li et al., 2025b; Diao et al., 2023; Chia et al., 2023)) have sought to improve CoT-based reasoning in LLMs, many models still tend to generate reasoning traces that are short, shallow, or lacking in depth. This remains a key bottleneck in addressing complex tasks that demand more elaborate and iterative reasoning. Recently, researchers have introduced a variety of reasoning models, such as OpenAI o1 (OpenAI, 2024), DeepSeek R1 (Guo et al., 2025), and Qwen QwQ (Team, 2025c), that are capable of generating long CoT solutions with strong self-validation and reflective reasoning patterns, enabling them to tackle complex reasoning tasks effectively.

Two primary approaches have proven effective in eliciting LLMs to generate long CoT reasoning traces: knowledge distillation and reinforcement learning (RL). The former approach leverages advanced models such as DeepSeek R1 to produce long CoT traces, which are then used to supervise smaller models through SFT. This distillation-based strategy has resulted in a series of compact yet competitive reasoning models (Ye et al., 2025; Moshkov et al., 2025; Muennighoff et al., 2025; Team, 2025a) that achieve performance comparable to DeepSeek R1. In the RL-based direction, reinforcement learning can be directly applied to base models or instruction-tuned models to enhance their reasoning capabilities. Several studies have successfully adopted this approach to reproduce reasoning models. For example, Yu et al. (Yu et al., 2025) introduced an open-source RL algorithm that successfully transformed the Qwen2.5-32B model into a reasoning-oriented version. Similarly, Zhang et al. (Zhang et al., 2025) proposed SRPO, a two-stage history-resampling policy optimization framework, which was also applied to Qwen2.5-32B and demonstrated superior performance over previous reasoning models with fewer training steps.

## 6.2. Distillation for Math Reasoning

Knowledge distillation from advanced models to smaller base models is an effective strategy for developing reasoning models that maintain strong performance while significantly reducing computational costs. DeepSeek (Guo et al., 2025) has released a series of distilled models that demonstrate exceptional performance in mathematical reasoning across various benchmarks, for example, DeepSeek-R1-Distill-Qwen-32B. Subsequent works have proposed to generate large-scale data from strong advanced models and then build mathematical reasoning models via distillation. Moshkov et al. (Moshkov et al., 2025) constructed a large-scale dataset containing 3.2M long-form reasoning solu-

tions based on 540,000 unique high-quality math problems. Their proposed model won the AI Mathematical Olympiad – Progress Prize 2. Similarly, Zhao et al. (Zhao et al., 2025) introduced a dataset of 1.4M question-response pairs accompanied by detailed reasoning traces. The models they developed outperformed the DeepSeek-R1-Distill-Llama-70B model across all math benchmarks. The Open R1 team (Team, 2025b) also released a dataset of 220K examples, along with a SFT model distilled from DeepSeek-R1. In contrast, Muennighoff et al. (Muennighoff et al., 2025) developed the s1-32B model using a much smaller dataset of only 1K questions paired with reasoning traces, yet surpassed o1-preview on competition-level math problems by up to 27%.

## 7. Conclusion and Future Work

In this work, we investigate how to induce reasoning behavior in a base language model based on different types of data. We first propose a simple and empirical definition of a reasoning model, based on the performance improvement of a base model after fine-tuning on a small number of high-quality reasoning traces, and demonstrate that as few as 20 reasoning examples can activate reasoning behavior in a base model. We then conduct extensive studies on two alternative data sources, including non-reasoning data and human-written solutions, and find that neither of them is sufficient to induce reasoning behavior in the base model, despite extensive prompt engineering and iterative refinement. Furthermore, we perform a series of ablation studies on the reasoning data and identify structural consistency in reasoning traces as a key factor in enabling the distillation of the reasoning ability.

For future work, one direction is to explore strategies that encourage deeper reasoning behaviors in LLMs, for example, providing partial solutions such as intermediate steps without final answers to prompt the model to complete the reasoning process on its own. We also plan to investigate other models and long CoT solutions to better understand model-specific reasoning behaviors. In addition, while this work focuses on mathematical problems, future studies could extend our framework to other reasoning-intensive domains such as symbolic logic, coding tasks, or scientific question answering, to examine the generalization of induced reasoning abilities. Finally, we aim to continuously refine human-written solutions by injecting more consistent reasoning patterns and harmonizing writing styles across annotators.

## References

Yew Ken Chia, Guizhen Chen, Luu Anh Tuan, Soujanya Poria, and Lidong Bing. Contrastive chain-of-thought prompting. *arXiv preprint arXiv:2311.09277*, 2023.



- Shizhe Diao, Pengcheng Wang, Yong Lin, Rui Pan, Xiang Liu, and Tong Zhang. Active prompting with chain-of-thought for large language models. *arXiv preprint arXiv:2302.12246*, 2023.
- 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 preprint arXiv:2501.12948*, 2025.
- Dacheng Li, Shiyi Cao, Tyler Griggs, Shu Liu, Xiangxi Mo, Eric Tang, Sumanth Hegde, Kourosh Hakhmaneshi, Shishir G Patil, Matei Zaharia, et al. LLMs can easily learn to reason from demonstrations structure, not content, is what matters! *arXiv preprint arXiv:2502.07374*, 2025a.
- Jia Li, Ge Li, Yongmin Li, and Zhi Jin. Structured chain-of-thought prompting for code generation. *ACM Transactions on Software Engineering and Methodology*, 34(2): 1–23, 2025b.
- Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019.
- Ivan Moshkov, Darragh Hanley, Ivan Sorokin, Shubham Toshniwal, Christof Henkel, Benedikt Schifferer, Wei Du, and Igor Gitman. AIMO-2 winning solution: Building state-of-the-art mathematical reasoning models with openmathreasoning dataset. *arXiv preprint arXiv:2504.16891*, 2025.
- 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. *arXiv preprint arXiv:2501.19393*, 2025.
- Mathematical Association of America. American Mathematics Competitions. <https://maa.org/student-programs/amc/>.
- Art of Problem Solving. AoPS Forums. <https://artofproblemsolving.com/community/>.
- OpenAI. OpenAI o1 preview. <https://openai.com/o1/>, 2024.
- Gerald Shen, Zhilin Wang, Olivier Delalleau, Jiaqi Zeng, Yi Dong, Daniel Egert, Shengyang Sun, Jimmy Zhang, Sahil Jain, Ali Taghibakhshi, et al. Nemo-aligner: Scalable toolkit for efficient model alignment. *arXiv preprint arXiv:2405.01481*, 2024.
- NovaSky Team. Sky-t1: Fully open-source reasoning model with o1-preview performance in \$450 budget. <https://novasky-ai.github.io/posts/sky-t1>, 2025a. Accessed: 2025-01-09.
- OpenR1 Team. Open r1 math 200k. <https://huggingface.co/datasets/open-r1/OpenR1-Math220k>, February 2025b. Accessed: 2025-05-19.
- Qwen Team. QwQ-32B: Embracing the Power of Reinforcement Learning, March 2025c. URL <https://qwenlm.github.io/blog/qwq-32b/>.
- Xuezhi Wang and Denny Zhou. Chain-of-thought reasoning without prompting. *arXiv preprint arXiv:2402.10200*, 2024.
- 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. *NeurIPS*, 2022.
- An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-Math Technical Report: Toward Mathematical Expert Model via Self-Improvement, 2024.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 Technical Report, 2025. URL <https://arxiv.org/abs/2412.15115>.
- Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*, 2025.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.
- Xiaojiang Zhang, Jinghui Wang, Zifei Cheng, Wenhao Zhuang, Zheng Lin, Minglei Zhang, Shaojie Wang, Yinghan Cui, Chao Wang, Junyi Peng, et al. Srpo: A cross-domain implementation of large-scale reinforcement learning on llm. *arXiv preprint arXiv:2504.14286*, 2025.
- Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*, 2022.

Han Zhao, Haotian Wang, Yiping Peng, Sitong Zhao, Xiaoyu Tian, Shuaiting Chen, Yunjie Ji, and Xiangang Li. 1.4 Million Open-Source Distilled Reasoning Dataset to Empower Large Language Model Training. *arXiv preprint arXiv:2503.19633*, 2025.

## A. Data Generation

### A.1. Data Generation with Instruction

#### Prompt: Data Generation with Instruction

I will give you a math problem and ask to solve it. Your goal is to write your thinking process as you approach this math problem.  
Solve the following mathematical problem in a natural, exploratory, and human-like reasoning style.  
Your reasoning should reflect how a person would think through the problem step by step, rather than providing a rigid, structured answer.  
Follow these guidelines:

1. Think aloud and reason through the problem dynamically:  
Start with intuitive observations and build from there.  
Use a conversational tone, as if you are thinking through the problem in real time.  
Include phrases like:  
"Hmm, let me see..."  
"Wait, does this actually work?"  
"Oh, I think I see a pattern!"
2. Incorporate self-doubt and self-correction:  
If you realize a mistake or reconsider an approach, explicitly acknowledge it.  
Say things like:  
"Wait, no, that can't be right. Let me check again."  
"Actually, I think I missed something there."  
"Hold on, let me backtrack and see where I went wrong."
3. Use trial and error before arriving at a final conclusion:  
Try different approaches and compare them.  
If an assumption is made, question whether it's valid.  
Example phrases:  
"Let's try assuming  $x=y=z$  and see what happens."  
"Hmm, this simplifies things, but does it actually lead to a solution?"
4. Verify and reflect on the final answer:  
Once you reach an answer, check if it actually satisfies the conditions.  
Explicitly validate by plugging values back in.  
Include phrases like:  
"Alright, let's plug this back in and see if it holds up."  
"Yep, this checks out seems like we got the right answer"  
"Wait, let's test it with a smaller case like  $n=2$  to make sure."
5. Make the reasoning process engaging and organic:  
Avoid overly rigid or textbook-style explanations.  
Feel free to pause, rethink, and approach things differently.  
Keep the explanation engaging, as if you're discussing it with a friend.

Finish the solution with **\*\*Final Answer\*\***  

$$\boxed{\text{{YOUR ANSWER}}}$$

{problem}

Please share your thinking process now. Remember: your goal is to write your thinking process as you approach this math problem. Note that we are not that much interested in the final solution, but much more in the process that lead you to it.  
Did you have references to other problems you solved before in your mind? Write

that down. Did you ponder upon which approach to take before doing derivations? Write that down. Does your current approach look too complicated and you need to try something else? Say that and write down any alternative ideas you have. Did you notice that something doesn't look good and you need to double check previous derivations for mistake? That's what we are looking for – write that down and correct the solution, but don't erase the mistake! The more detailed this is, the better.

## A.2. Data Generation with Few-Shot Instruction

### Prompt: Data Generation with Few-Shot Instruction

I will give you a math problem and ask to solve it. Your goal is to write your thinking process as you approach this math problem.

Solve the following mathematical problem in a natural, exploratory, and human-like reasoning style.

Your reasoning should reflect how a person would think through the problem step by step, rather than providing a rigid, structured answer.

Follow these guidelines:

1. Think aloud and reason through the problem dynamically:  
Start with intuitive observations and build from there.  
Use a conversational tone, as if you are thinking through the problem in real time.  
Include phrases like:  
"Hmm, let me see..."  
"Wait, does this actually work?"  
"Oh, I think I see a pattern!"
2. Incorporate self-doubt and self-correction:  
If you realize a mistake or reconsider an approach, explicitly acknowledge it.  
Say things like:  
"Wait, no, that can't be right. Let me check again."  
"Actually, I think I missed something there."  
"Hold on, let me backtrack and see where I went wrong."
3. Use trial and error before arriving at a final conclusion:  
Try different approaches and compare them.  
If an assumption is made, question whether it's valid.  
Example phrases:  
"Let's try assuming  $x=y=z$  and see what happens."  
"Hmm, this simplifies things, but does it actually lead to a solution?"
4. Verify and reflect on the final answer:  
Once you reach an answer, check if it actually satisfies the conditions.  
Explicitly validate by plugging values back in.  
Include phrases like:  
"Alright, let's plug this back in and see if it holds up."  
"Yep, this checks out, seems like we got the right answer!"  
"Wait, let's test it with a smaller case like  $n=2$  to make sure."
5. Make the reasoning process engaging and organic:  
Avoid overly rigid or textbook-style explanations.  
Feel free to pause, rethink, and approach things differently.  
Keep the explanation engaging, as if you're discussing it with a friend.

Finish the solution with **\*\*Final Answer\*\***



\[ \boxed{{YOUR ANSWER}} \]

I first provide you with the following examples of how to solve math problems and then provide you a problem you need to solve.

{examples}

This is the problem you need to solve:  
{problem}

Please share your thinking process now. Remember: your goal is to write your thinking process as you approach this math problem. Note that we are not that much interested in the final solution, but much more in the process that lead you to it.

Did you have references to other problems you solved before in your mind? Write that down. Did you ponder upon which approach to take before doing derivations? Write that down. Does your current approach look too complicated and you need to try something else?

Say that and write down any alternative ideas you have. Did you notice that something doesn't look good and you need to double check previous derivations for mistake? That's what we are looking for – write that down and correct the solution, but don't erase the mistake!

The more detailed this is, the better.

## B. Data Edition

### B.1. Data Edit with Instruction

#### Prompt: Data Edition with Instruction

You will be provided with a mathematical solution.  
Your task is to revise it by weaving in self-questioning, step-by-step verification, and natural self-correction.  
Keep the original content and structure intact, but enhance the reasoning process with thoughtful self-reflection.

Instructions:

1. Preserve the format and logical flow of the original solution.  
Do not add extra sections or headers that might interfere with smooth integration.
2. Incorporate at least 3 – 5 self verification moments throughout the reasoning.  
Use natural phrasing such as:

Wait, something seems off here, let me double check.

Hold on, did I consider all the possible cases?

Let's go back and verify this step before moving on.

3. Stay aligned with the original solution, ensuring a seamless blend with the surrounding content.

Now, I will provide you with a solution. Please enhance it by incorporating self-questioning, verification, and natural self-correction to ensure both rigor and clarity.

Solution: {solution}

## B.2. Data Edit with Few-Shot Instruction

### Prompt: Data Edit with Few-Shot Instruction

You will be provided with a mathematical solution.

Your task is to revise it by weaving in self-questioning, step-by-step verification, and natural self-correction.

Keep the original content and structure intact, but enhance the reasoning process with thoughtful self-reflection.

Instructions:

1. Preserve the format and logical flow of the original solution.  
Do not add extra sections or headers that might interfere with smooth integration.
2. Incorporate at least 35 self verification moments throughout the reasoning.  
Use natural phrasing such as:

Wait, something seems off here, let me double check.

Hold on, did I consider all the possible cases?

Let's go back and verify this step before moving on.

3. Stay aligned with the original solution, ensuring a seamless blend with the surrounding content.

I will provide you with a few examples.

Please refer to these examples as template and follow the instructions above when revising the solution I give you afterward.

{examples}

Now, I will provide you with a solution. Please enhance it by incorporating self-questioning, verification, and natural self-correction to ensure both rigor and clarity.

Solution: {solution}

## B.3. Data Edit with Incorrect Solution Combination

### Prompt: Data Edit with Incorrect Solution Combination

You are an advanced mathematical reasoning assistant.

Your goal is to merge a correct solution while incorporating elements from an incorrect solution to create a cohesive, logically structured response that includes natural self-corrections.

The final answer must remain correct while the reasoning process should be enhanced

through logical self-reflection and error correction.  
 please start with your modified output with "Solution"  
 please make sure finish the solution with **\*\*Final Answer\*\***  
`\[ \boxed{{YOUR ANSWER}} \]`

Instructions:

1. Analyze the multiple correct solutions and consolidate them into a single refined version that preserves all key insights.
2. Identify mistakes from the incorrect solutions (e.g., calculation errors, conceptual misunderstandings, missing steps).
3. Inject 3 – 6 subtle mistakes from the incorrect solutions into the reasoning process.
4. Ensure these mistakes are discovered and corrected naturally using logical problem-solving (not explicitly stated as intentional errors).
5. Use a natural and engaging style for self-correction, incorporating phrases like:  
 Wait, something feels off here... let me check.  
 "Oh! I think I see the mistake..."  
 "Hold on, I might have miscalculated this part."  
 Let's go back and check this, something doesn't look right.
6. Maintain clarity, logical flow, and rigor in the final solution.
7. Ensure the final answer is correct, even after introducing and correcting the errors.

Please avoid using phrases like "let us introduce a few mistakes," "we will intentionally add some errors," or "for demonstration purposes, we will insert some mistakes," and ensure the wording sounds natural.

Now, I will provide one correct solution and one incorrect solution. Please merge them into a single solution following the instructions above.  
 Ensure that the final response integrates insights from the correct solutions while incorporating and naturally correcting errors from the incorrect solution through logical self-correction

Correct solutions:

Correct solution : {correct}  
 Incorrect solution : {incorrect}

### C. SFT Accuracy for Qwen2.5-7B and Qwen2.5-14B

We use the same data as in Section 2.4 and conduct SFT on Qwen2.5-7B and Qwen2.5-14B, using the same training and evaluation settings as Qwen2.5-32B. The SFT accuracy results are shown in Figure 2. The results show that Qwen2.5-7B performs below the baseline, while Qwen2.5-14B only slightly outperforms it. However, unlike Qwen2.5-32B, neither model is able to effectively enter reasoning mode with only minimal supervision and a limited number of high-quality, long CoT examples.

### D. SFT Accuracy under Hyperparameters

We performed a grid search for hyperparameter tuning on Qwen2.5-32B using 50 reasoning examples, with the results summarized in Table 3. Since we found that a learning rate of  $1e-5$  and a step size of 50 yielded the best performance, we

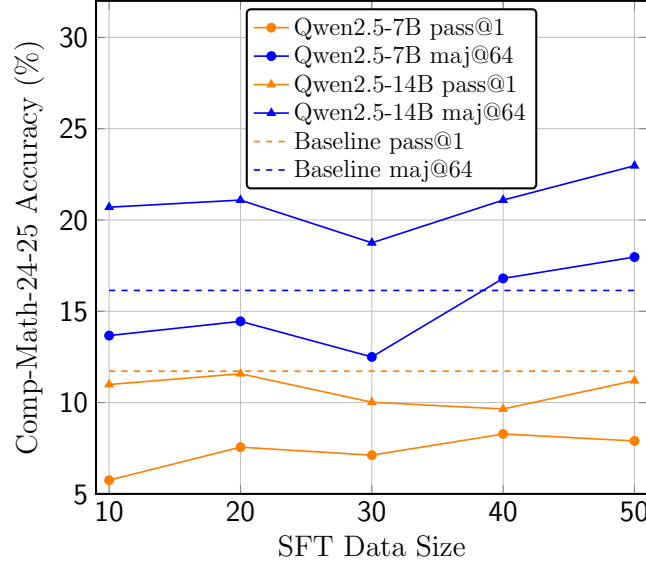


Figure 2. SFT accuracy of Qwen2.5-7B and Qwen2.5-14B on Comp-Math-24-25 under varying data sizes.

fixed these hyperparameters for all experiments presented in this paper.

Hyperparameters	pass@1	maj@64
LR = 1e-4, Steps = 50	6.52%	12.50%
LR = 1e-5, Steps = 50	19.29%	33.59%
LR = 1e-6, Steps = 50	8.53%	18.36%
LR = 1e-5, Steps = 100	18.60%	31.64%
LR = 1e-5, Steps = 200	15.05%	32.42%

Table 3. SFT accuracy on Comp-Math-24-25 under varying hyperparameters.