LONG IS MORE IMPORTANT THAN DIFFICULT FOR TRAINING REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Difficult problems, which often produce longer reasoning traces, are widely regarded as key drivers for enhancing the performance of reasoning models. In this work, we challenge this coupled assumption by disentangling problem difficulty and reasoning length, and demonstrate that *reasoning length* itself plays the dominant role. We introduce a simple yet effective method to synthetically construct long-chain reasoning data without requiring inherently challenging tasks, leading to the Long1K dataset, comprising only 1,000 training samples. Fine-tuning on Long1K produces Long1K-32B, which achieves state-of-the-art results on benchmarks such as MATH500 (95.6%) and GPQA Diamond (71.1%), outperforming models trained on vastly larger datasets, including DeepSeek-R1-Distill-Qwen-32B, with improvements of 1.3% and 9% respectively. Further analysis shows that longer reasoning sequences promote more structured reasoning, improve long-range instruction following, and achieve superior scaling efficiency compared to inference-only strategies. Our findings establish reasoning length as a critical and independent scaling axis for enhancing the reasoning capabilities of large language models. The model, code, and dataset are all open-sourced, available at `https://anonymous.4open.science/r/LONG1k-32B`.

1 INTRODUCTION

Difficult problems, which often induce longer reasoning traces, are widely regarded as crucial for improving the performance of large language model (LLM)-based reasoning systems. Recent models such as ChatGPT o1 Jaech et al. (2024); OpenAI (2024), DeepSeek-R1 Guo et al. (2025), and Gemini 1.5 Kavukcuoglu (2025) demonstrate that producing extended sequences of reasoning is critical for achieving high accuracy on complex tasks.

The prevailing assumption in reasoning model training is that increasing problem difficulty leads to deeper reasoning chains, thereby enabling better exploitation of the model's pre-trained knowledge Min et al. (2024); Ye et al. (2025); Muennighoff et al. (2025). Following this intuition, recent efforts such as STILL-2 Min et al. (2024), DRT-o1 Wang et al. (2024a), and LIMO Ye et al. (2025) emphasize carefully curating highly challenging datasets to drive reasoning improvement. However, truly difficult problems are scarce in real-world corpora: for instance, only 2.35% of the Sky-T1-32B-Preview Li et al. (2025a;b) and 7.34% of the LIMO Ye et al. (2025) training sets exceed 15k tokens in reasoning length. This scarcity inherently limits the scalability of difficulty-driven approaches.

Despite the strong correlation between difficulty and reasoning length, it remains unclear whether difficulty itself is the key factor — or whether *reasoning length* alone is sufficient to drive model improvements. In this work, we disentangle these two factors through controlled experiments and show that *reasoning length* is the dominant axis for scaling reasoning model performance.

To systematically explore this hypothesis, we propose a simple yet effective method to synthetically generate long-chain reasoning data without requiring intrinsically difficult problems. Using this approach, we construct the **Long1K** dataset, comprising 1,000 training samples and most of their reasoning sequences extending up to 32k tokens. Fine-tuning a Qwen2.5-32B-Instruct model on Long1K yields **Long1K-32B**, which achieves state-of-the-art results on MATH500 Hendrycks et al. (2021)(95.6%) and GPQA Diamond Rein et al. (2023)(71.1%), outperforming models trained on vastly larger datasets such as DeepSeek-R1-Distill-Qwen-32B by 1.3% and 9% respectively.

Our contributions are summarized as follows:

- **Challenging a dominant assumption:** We demonstrate through controlled experiments that reasoning length, rather than problem difficulty, is the key factor in training effective reasoning models.
- **Identifying a scaling law on reasoning length:** We show that model performance improves nearly linearly with the logarithm of reasoning trace length, highlighting reasoning length as a new scaling dimension.
- **Developing a simple synthesis strategy:** We propose an efficient method to generate arbitrarily long reasoning sequences, releasing the Long1K dataset and Long1K-32B model to support further research.
- **Providing new insights into reasoning model behavior:** We conduct in-depth analysis showing that longer training reasoning sequences improve structural coherence, enhance instruction-following ability across long contexts, and yield more efficient scaling than inference-only strategies.

2 Reasoning Length is More Important than Difficulty

Conventional wisdom holds that difficult problems are essential for training reasoning models Lightman et al. (2023). However, because complex problems often require more extensive reasoning traces, it remains unclear whether problem difficulty or reasoning length is the primary driver of model performance. In this section, we disentangle these two factors by designing controlled experiments that (1) fix the difficulty level while varying the reasoning length, and (2) fix the reasoning length while varying the difficulty level. Further, we scaling reasoning length and observe the performance consistently improves.

2.1 Preliminaries

We conduct our experiments on two high-quality reasoning datasets: Openthoughts-114k Team (2025b) and S1K-1.1 Muennighoff et al. (2025), both derived from DeepSeek-R1 to ensure fair comparisons. Our primary base model is Qwen2.5-32B-Instruct Yang et al. (2024), selected for its strong performance on reasoning tasks. All models are trained using full-parameter fine-tuning on 8×A800-80GB GPUs, with a learning rate of 2e-4 and cosine decay scheduling. Unless otherwise specified, we use 3 epochs and a total batch size of 16. Evaluation is based on final answer accuracy, using Qwen2.5-72B-Instruct Yang et al. (2024) as an external verifier. Further training details and benchmark descriptions are provided in Appendix A.

2.2 Control Difficulty and Vary Reasoning Length

To investigate whether increasing reasoning length can boost performance, we first control *problem difficulty* by ensuring all datasets contain the exact same set of problems. Specifically, we sampled 500 problems from Openthoughts-114k Team (2025b), and used DeepSeek-R1-Distill-Qwen-32B Guo et al. (2025) to generate 10 solutions for each problem. We filtered out incorrect solutions and randomly retained 4 valid ones per problem. These valid solutions were then sorted by token length and grouped into four datasets: for each problem, the shortest correct solution went into Set 1, the second shortest into Set 2, and so on, with Set 4 containing the longest.

We fine-tuned Qwen2.5-32B-Instruct on each of these four datasets separately and evaluated the resulting models on the MATH500 and AIME24 benchmarks. As illustrated in Table 1, model accuracy consistently improves as the average length of the reasoning traces increases. For instance, moving from solutions averaging 2588 tokens to those averaging 5986 tokens yielded about an 5.6% improvement on MATH500 and 11.1% improvement on AIME2024, respectively.

Discussion After controlling for problem difficulty, longer reasoning traces still produce better results. Although Wu et al. (2025) found that excessively long solutions could hurt performance in narrowly defined arithmetic tasks, our experiments on larger, real-world datasets show that longer reasoning consistently enhances accuracy, indicating that extended reasoning is more likely to be beneficial in broader or more diverse domains.

Table 1: Performance of Qwen2.5-32B-Instruct after being trained on datasets containing identical problems but differing in reasoning-trace length. Longer solutions lead to higher accuracy.

Dataset	Avg. length	MATH500	AIME2024
Set 1	2588	86.0%	30.0%
Set 2	3425	87.2%	31.1%
Set 3	4355	87.8%	33.3%
Set 4	5986	91.6%	41.1%

2.3 Control Reasoning Length and Vary Difficulty

In this subsection, we hold reasoning length roughly constant but vary problem difficulty. To achieve this, we design datasets where "easier" problems are made longer by adding multiple sub-questions, whereas "difficult" problems are inherently complex but contain only a single question. This ensures both sets have similar token lengths in their solution traces but differ in intrinsic difficulty.

2.3.1 Setup

We create two comparison groups to control for data quality and diversity Lightman et al. (2023); Muennighoff et al. (2025), factors known to influence model training:

Group 1: Synthetic Composite Problems From math datasets Team (2025a), we first extract a pool of core mathematical concepts using Qwen2.5-72B-Instruct and then randomly combine exactly three concepts to form two categories of problems with DeepSeek-R1-Distill-Qwen-32B:

- **Difficult (Composite) Problems:** Each integrates multiple concepts into one challenging question.
- **Easy (Composite) Problems:** Covering the same concepts, but broken into several simpler sub-questions.

To ensure correctness and comparable average reasoning lengths across both categories, we again sample and filter solutions using DeepSeek-R1-Distill-Qwen-32B. Because they draw on the same set of concepts and follow a similar synthetic procedure, these two problem sets have comparable topic coverage and data quality. Detailed composition methods are provided in Appendix B.

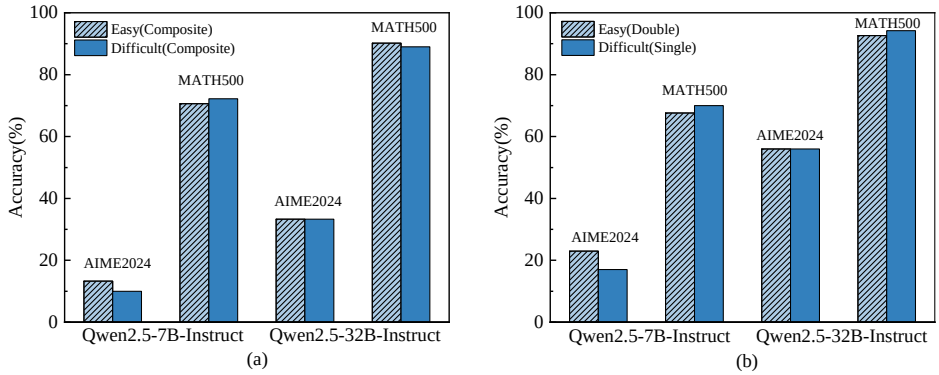

Figure 1: (a) Group 1: accuracy comparison of models after training on Easy Problems(Composite) and Difficult Problems(Composite). (b) Group 2: accuracy comparison of models after training on Easy Problems(Double) and Difficult Problems(Single).

Group 2: Original Problems from Openthoughts-114k In this setting, we select "difficult" single problems directly from Openthoughts-114k as our **Difficult (Single)** set. Each problem is inherently more conceptually challenging, and its corresponding solution is typically longer. By contrast, our **Easy (Double)** set is formed by concatenating pairs of shorter and simpler problems (along with their solutions) into a single item, ensuring that both sets end up having comparable token lengths in their reasoning traces. This approach uses naturally occurring items for the difficult set and a

straightforward concatenation strategy to lengthen the easier set, although it does not strictly control for problem diversity.

We use the Qwen2.5-72B-Instruct model to assess group difficulty through direct evaluation (rating question difficulty on a 1–10 scale Team (2025a)) and the pass@5 method. Both metrics consistently confirm effective difficulty differentiation across sets. Appendix C shows representative examples of the four categories with results from both evaluation methods. Despite similar solution trace lengths, the Easy (Double) and Easy (Composite) datasets receive lower difficulty levels in their respective comparison groups.

2.3.2 Results

We train Qwen2.5-7B-Instruct and Qwen2.5-32B-Instruct on each group and the results are shown in Figure 1. Across both Group 1 and Group 2, performance on standard math benchmarks (e.g., AIME2024, MATH500) remains largely comparable between the "easier" and "difficult" sets—even though the latter have higher conceptual complexity. This reinforces our finding that reasoning length has a more pronounced effect on model accuracy than inherent problem difficulty.

2.4 Scaling Reasoning Length of Training Data

Since reasoning length has emerged as a key factor in model performance, a natural question arises: **How does increasing the reasoning length of training data affect performance at scale?** To investigate this, we examine how variations in the average number of reasoning tokens per sample influence final accuracy.

We randomly sampled several datasets from Openthoughts-114k, each containing 500 problems with average reasoning lengths ranging from 1.5k to 12k tokens. Each dataset was used to fine-tune the Qwen2.5-32B-Instruct model, and the resulting models were evaluated on the MATH500 and GPQA Diamond benchmarks. The results are shown in Figure 2.

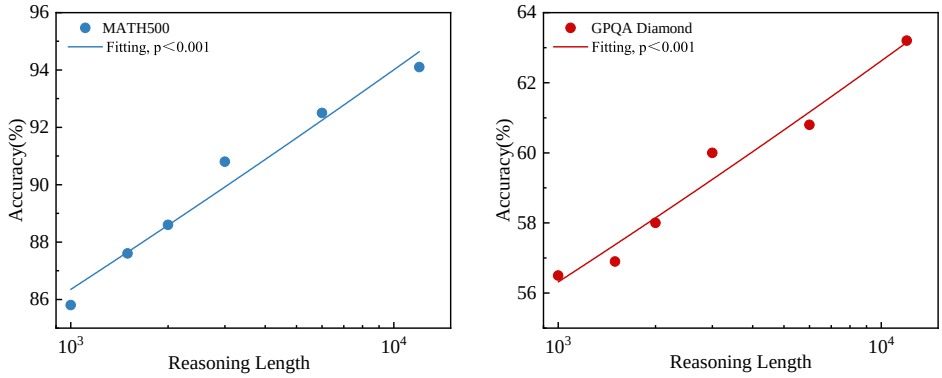

Figure 2: Performance on (a) MATH500 and (b) GPQA Diamond as a function of average reasoning length per training problem. The x-axis is log-scaled; performance follows a power-law trend.

We observe that across both benchmarks, as the reasoning length of the training data increases, model performance consistently improves. When plotted on a log-log scale, the relationship appears nearly linear, suggesting potential power-law behavior. After log-transforming the data, we fit a power-law model using ordinary least squares (OLS) regression. Both fits show highly significant scaling behavior ($p < 0.001$), confirming a strong and statistically significant correlation between reasoning length and accuracy. For detailed analysis and fitted equations, please refer to the Appendix D.

3 Create a Extremely Long Reasoning Dataset

3.1 Synthesizing Reasoning Data of Arbitrary Length

Natural, lengthy reasoning data are extremely scarce in real-world scenarios. We therefore propose a simple method to create arbitrarily long reasoning examples by concatenating multiple original problems into a single user prompt, and merging their respective thoughts and solutions into a single

output. A similar approach was already employed in constructing the *Easy (Double)* dataset in Section 2. Table 2 shows the prompt format we use.

Table 2: Prompts and output format for synthetic data. **PROBLEM-i**, **THINK-i**, and **SOLUTION-i** each come from a single problem sampled from a reasoning dataset (e.g., Openthoughts-114k). The underlined texts are templates used to link problems, thoughts, and solutions. We ask GPT-4o to produce multiple paraphrases of these templates for diversity Taori et al. (2023).

User Prompt	<|im_start|>user\n I need help with the following problems. The first one is **PROBLEM-1**, the second one is **PROBLEM-2** ... , and the final one is **PROBLEM-N**. <|im_end|>\n
Output	<|im_start|>system\n <|begin_of_thought|>\n\n I will handle these problems one by one. I will start with the first problem. **THINK-1**.\n Now I will turn to the second problem. **THINK-2**. ... OK, now is the last problem. **THINK-N** <|end_of_thought|>\n\n <|begin_of_solution|>\n\n The solution for the first problem is as belows. **SOLUTION-1**.\n The solution for the second problem is as belows. **SOLUTION-2**. ... The solution for the last problem is as belows. **SOLUTION-N**.<|end_of_solution|> <|im_end|>

This design offers several advantages. First, we can easily generate arbitrarily long data by concatenating additional problems and solutions. Second, the structure aligns with the typical style of current reasoning models, which tend to formulate the entire chain of thought before presenting the final solution. This allows our synthetic data to be directly combined with other original datasets. Finally, the long-range dependencies in our concatenated format encourage the model to reason consistently across extended contexts. To generate **SOLUTION-N**, the model must locate **PROBLEM-N** at the beginning, follow **THINK-N** in the middle, and ignore unrelated information, demonstrating its ability to focus on relevant parts within a long context.

3.2 The Long1K Dataset

We now describe the construction of the *Long1K* dataset, which serves as the primary training resource for our final model. Long1K is composed of data from two main sources: Openthoughts-114k Team (2025b) and S1K-1.1 Muennighoff et al. (2025), both annotated using DeepSeek-R1.

The dataset contains a total of 1,000 problem-solution pairs. Among them, 800 samples are synthesized by randomly selecting and pairing two problems from Openthoughts-114k. Their problem statements, reasoning traces, and final answers are concatenated into a single long-form instance, yielding an average total sequence length of approximately 32k tokens. To mitigate overfitting to the format or linguistic patterns of such concatenated examples, we additionally include 200 original, non-concatenated problems randomly sampled from S1K-1.1. This portion of the dataset introduces greater diversity in problem format and helps improve model robustness.

Unlike prior works that rely heavily on manual data curation—often filtering for highly diverse or extremely difficult problems—we adopt a much simpler data construction strategy: the 800 long examples are generated via purely random sampling and automatic concatenation. We regard this simplicity and scalability as one of the key strengths of our approach.

All training examples are thoroughly decontaminated against our evaluation sets (MATH500, GPQA Diamond, AIME2024, and AIME2025) using 8-gram overlap filtering, and duplicate entries are removed to ensure fairness in evaluation.

4 Results

4.1 Setup

We fine-tuned Qwen2.5-32B-Instruct on the Long1K dataset to obtain the final model, Long1K-32B, with a maximum output length of 32k tokens. It was compared against several strong baselines,

including S1-32B, S1.1-32B, LIMO, OpenThinker-32B, and DeepSeek-R1-Distill-Qwen-32B. All models are fine-tuned on Qwen2.5-32B-Instruct for consistency. Except for S1-32B, all baselines are trained on reasoning traces distilled using DeepSeek-R1. This consistent setup ensures a fair comparison, allowing us to independently assess the benefit brought by *longer reasoning traces* in the Long1K dataset. More details on the baseline models can be found in Appendix F.

4.2 Main Results

Our experimental results, summarized in Table 3, highlight the strong performance of Long1K-32B across multiple benchmarks.

- **Comparable-Size Models:** Among models trained on similarly sized datasets, Long1K-32B achieves substantial gains across most benchmarks. For instance, while LIMO represents the strongest baseline in this category, it lags behind Long1K-32B by 4.0% on AIME2025 and 4.4% on GPQA Diamond.
- **Large-Scale Models:** Despite being trained on a much smaller dataset, Long1K-32B achieves performance comparable to large-scale models. Notably, it outperforms DeepSeek-R1-Distill-Qwen-32B on two of the four benchmarks, including a 9.0% gain on GPQA Diamond.

One exception is AIME2024, where Long1K-32B underperforms compared to other models. We attribute this to limited data diversity in our training set, as Long1K was constructed without intensive data filtering or selection for coverage. Additionally, the AIME2024 benchmark consists of only 30 questions, making the evaluation potentially sensitive to dataset-specific variance.

Notably, all problems and solutions in Long1K-32B are randomly sampled from the same data pool as S1.1-32B and OpenThinker-32B. Yet, Long1K-32B achieves significantly better performance, emphasizing the advantage of longer reasoning sequences.

These findings suggest that extending reasoning length through simple concatenation is a more scalable and effective strategy than manually curating difficult problems. Optimizing reasoning depth, not difficulty, offers a practical path for improving model capability.

Table 3: Performance comparison of different models across multiple reasoning benchmarks (pass@1). The best results for each benchmark are highlighted in bold, with the second-best underlined. The results for S1-32B and S1.1-32B do not use budget forcing.

Model Name	Dataset Size	MATH500	AIME2024	AIME2025	GPQA Diamond
S1-32B	1k	92.6	50.0	26.7	56.6
S1.1-32B	1k	87.4	59.3	42.7	62.0
LIMO	0.8k	<u>94.8</u>	57.1	49.3	<u>66.7</u>
OpenThinker-32B	114k	90.6	<u>68.0</u>	49.3	63.5
DeepSeek-R1-Distill-Qwen-32B	800k	94.3	**72.6**	**55.9**	62.1
Long1K-32B	1k	**95.6** $\pm^{1.0}_{1.6}$	50.7 $\pm^{6.0}_{4.0}$	<u>53.3</u> $\pm^{10.0}_{10.0}$	**71.1** $\pm^{2.1}_{2.4}$

5 Why Longer Reasoning Data Build Better Models

Recent studies have shown that longer training contexts can substantially benefit large language models. Prolong Gao et al. (2024) found that pretraining with 512k-token sequences leads to stronger models than training with 64k-token contexts. Similarly, Jin et al. Jin et al. (2024) reported that even noisy or partially incorrect rationales can improve model outcomes—provided that the inference traces are sufficiently long. While these results support the intuition that "longer is better," they neither target reasoning models explicitly nor fully explain the underlying mechanisms.

In this section, we offer a deeper empirical analysis focused on *reasoning models*, i.e., LLMs trained to produce extended chains of thought before concluding with an answer (such as OpenAI's `o1` and

o3). Our results reveal not only practical benefits, but also new insights into how training length impacts model structure and control.

5.1 Observation 1: Structured Failures

Training with longer reasoning sequences improves not just answer accuracy, but also the structure of model failures.

Previous studies Yamin et al. (2024) observed that performance on long reasoning tasks often degrades due to increased reflection and backtracking, a phenomenon related to the "lost in the middle" effect Liu et al. (2024). To assess whether long-sequence training mitigates this instability, we compare two models trained on datasets with average reasoning lengths of 1.5k and 12k tokens, respectively. Following Li et al. (2025c); Ye et al. (2025), we track the usage of transition words such as "but" and "wait," which signal re-evaluation or correction during inference.

As shown in Table 4, while both models use such transitions sparingly (2%) in successful cases, the 1.5k-trained model's transition rate surges to nearly 9% in failed cases. In contrast, the 12k-trained model maintains a stable transition frequency (2%), suggesting that even when producing incorrect answers, its reasoning remains more internally coherent.

Table 4: Transition word frequencies in successful and failed cases. Longer training traces yield more structured reasoning even when errors occur.

Avg. Train Length	Outcome	Top-10 Frequent Words (Frequency)
1.5k	Correct	so(1.98%), and(1.25%), let(1.08%), **wait**(1.07%), **but**(0.91%), me(0,60%), therefore(0.56%), which(0.47%), we(0.47%), if(0.46%)
12k	Correct	so(1.83%), and(1.19%), **but**(1.19%), let(0.93%), **wait**(0.81%), therefore(0.65%), we(0.59%), if(0.52%), which(0.50%), me(0.45%)
1.5k	Wrong	**but**(5.05%), **wait**(3.78%), so(1.26%), therefore(1.16%), and(1.01%), angle(0.69%), let(0.63%), sqrt(0.60%), maybe(0.55%), sum(0.49%)
12k	Wrong	and(1.42%), **but**(1.27%), so(1.08%), **wait**(0.80%), angle(0.71%), let(0.65%), we(0.60%), therefore(0.59%), if(0.52%), sqrt(0.51%)

5.2 Observation 2: Improved Long-Range Instruction Following

Reasoning models often struggle more than their base models to maintain instruction-following ability[1]. We hypothesize that training with longer contexts improves this ability by forcing models to retain and refer back to instructions issued far earlier.

To evaluate this, we compared Long1K-32B and S1.1-32B on the MATH500 benchmark, adding a requirement for each response to conclude with "After finishing, say 'I have done that."'—a pattern unseen during training.

We group samples by response length into four bins. As Table 5 shows, S1.1-32B's instruction-following rate drops from 96% (short responses) to 52.8% (longest responses), whereas Long1K-32B maintains nearly perfect adherence until 5k tokens and achieves 81.6% even for 20k-token responses.

Moreover, even on short contexts (1-2k tokens), Long1K-32B exhibits better compliance than S1.1-32B (whose average response length in training data is 9k), supporting prior observations Gao et al. (2024) that training on longer contexts benefits shorter-context tasks as well.

To systematically evaluate Long1K-32B's advantage in instruction-following capability, we conducted comprehensive testing on the MathIF benchmark Fu et al. (2025) , with the results presented in Appendix G. In terms of the comprehensive metric HAcc (which reflects both mathematical and instruction-following abilities), Long1K-32B significantly outperformed S1.1-32B in 14 out of 15 test sets. For the SAcc metric (which solely measures instruction-following ability), Long1K-32B also demonstrated superior performance across all test sets compared to S1.1-32B.

[1] https://lmarena.ai/?leaderboard

Table 5: Instruction-following performance across different response lengths.

Group	S1.1-32B			Long1K-32B		
	Instr. Follow(%)	Acc.(%)	Avg. Length	Instr. Follow(%)	Acc.(%)	Avg. Length
1	96.0	100	1.1k	100	99.2	1.2k
2	92.8	98.4	2.0k	100	100	2.5k
3	84.0	100	3.3k	99.2	96.8	5.0k
4	52.8	70.4	9.3k	81.6	74.4	19.7k

5.3 Observation 3: Training-Length Scaling Outperforms Inference-Length Scaling

One might suspect that Long1K's gains come merely from producing longer outputs at test time. However, Figure 3 refutes this.

Even with 32k-token training sequences, the average inference output length remains 7k tokens (Figure 3(a)). Moreover, accuracy improves *linearly* with inference length *only when training length increases* (Figure 3(b)). By contrast, prior works Muennighoff et al. (2025); Bi et al. (2024) needed to increase inference budgets exponentially to achieve linear performance gains.

Thus, training-length scaling is more efficient and impactful than inference-only scaling.

5.4 Summary and Broader Implications

Training on longer reasoning traces brings three distinct benefits. Firstly, errors remain coherent, not chaotic. Secondly, models better retain directives over long contexts. Finally, training-length scaling is more efficient than inference-only scaling. Importantly, our experiments in Section 2 demonstrate that these benefits stem from increased reasoning length itself, rather than from greater problem difficulty. This suggests that *reasoning length constitutes an independent scaling axis* for advancing reasoning models—complementary to model size and training data volume.

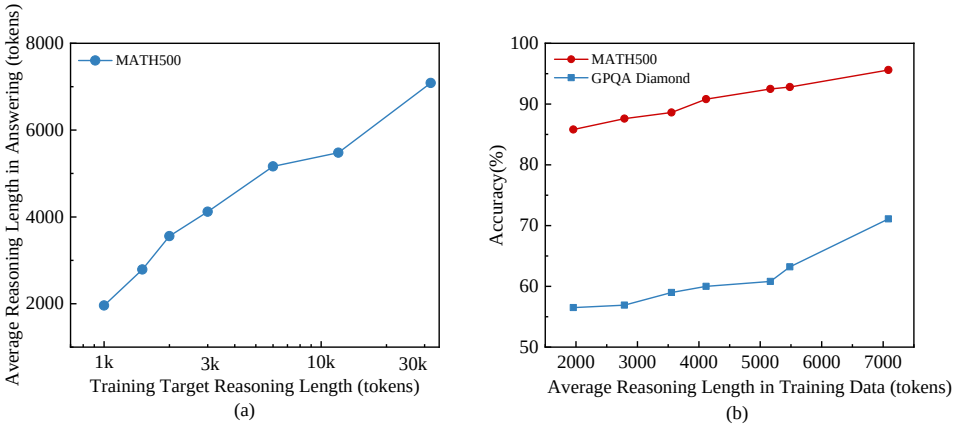

Figure 3: (a) Average reasoning length in answer vs. training target reasoning length. (b) Model accuracy vs. average reasoning length in answer.

6 Related Work

6.1 Reasoning Models and Long-Chain Reasoning

The development of long-chain reasoning models has attracted increasing attention as large language models (LLMs) scale up. Early observations Brown et al. (2020); Wei et al. (2022) noted that larger LLMs naturally exhibit better multi-step reasoning abilities. Recent systems such as DeepSeek-R1 Guo et al. (2025), ChatGPT o1 Jaech et al. (2024), Gemini 1.5 Kavukcuoglu (2025), and QwQ-32B Team (2025c)specifically optimize for long-form reasoning through distillation or reinforcement learning. Distilled models such as DeepSeek-R1-Distill-Qwen-32B and S1.1-32B Muennighoff et al. (2025) further demonstrate that strong reasoning performance can be achieved by carefully designing

reasoning traces and distilling them into smaller backbones. However, existing works often assume that the difficulty of problems is crucial for training high-quality reasoning models, without isolating the role of reasoning sequence length itself.

6.2 Challenges in Reasoning Training: Difficulty vs. Length

Several recent studies highlight the importance of difficult, complex problems for training reasoning models Muennighoff et al. (2025); Ye et al. (2025); Ding et al. (2024); Luo et al. (2023). These datasets typically select problems by difficulty, diversity, and depth, assuming harder tasks yield stronger reasoning skills. In contrast, preliminary findings Jin et al. (2024); Gao et al. (2024) suggest that even noisy or imperfect reasoning traces can help if sufficiently long, pointing to reasoning length as an independent scaling factor. Building on this, we disentangle problem difficulty from reasoning length and show that length alone explains most performance gains, without requiring inherently challenging problems.

6.3 Scaling Laws in Reasoning and Inference Efficiency

Scaling laws for LLMs Kaplan et al. (2020); Hoffmann et al. (2022); Snell et al. (2024); Ballon et al. (2025); Shao et al. (2024) have established predictable relationships between model size, data volume, and performance. More recently, scaling along the inference budget—allocating more tokens during inference—has been explored as a way to improve downstream reasoning Bi et al. (2024); Muennighoff et al. (2025). However, inference-time scaling typically incurs exponential cost to achieve linear performance gains. Our study proposes training-time scaling along the dimension of reasoning length, showing that longer reasoning traces during training yield more efficient scaling behavior and better generalization, particularly in complex multi-step reasoning tasks.

6.4 Comparison with Data Packing

While the data construction method in this study bears formal resemblance to data packing Wang et al. (2024b), they are fundamentally distinct. Data packing aims to improve training efficiency by concatenating independent samples to reduce padding tokens, whereas our method focuses on constructing coherent long sequences to disentangle the effects of problem difficulty and reasoning length, thereby validating that purely increasing reasoning steps enhances model capability. Experimental results demonstrate that our approach significantly improves model reasoning performance (e.g., 1K samples outperforming the 114K baseline), rather than merely optimizing computational efficiency.

7 Conclusion

This paper revisits a fundamental assumption in reasoning model training: whether problem difficulty or reasoning length plays a more critical role in performance. While previous studies often attributed improvements to task difficulty, our experiments suggest that it is the length of reasoning traces, rather than the intrinsic challenge of the problems, that primarily drives model gains.

To investigate this, we propose a simple yet effective method to extend reasoning sequences without requiring harder tasks, leading to the construction of the Long1K dataset. Fine-tuning on Long1K produces Long1K-32B, a model that achieves state-of-the-art results on MATH500 and GPQA Diamond, outperforming baselines trained with comparable or even significantly larger data.

Beyond empirical results, our analysis reveals that longer reasoning sequences strengthen the model's internal reasoning structure, improve long-range instruction following, and offer more efficient scaling than inference-only methods. These findings highlight reasoning length as a distinct and powerful scaling dimension for building stronger reasoning models.

Looking ahead, optimizing reasoning length during training, balancing reasoning complexity with efficiency, and designing curriculum strategies for progressive scaling offer promising directions for advancing reasoning-centric LLMs. We hope this study encourages future work to prioritize the structure and depth of reasoning over simply increasing task difficulty.

Ethics Statement

This research adheres to the ICLR Code of Ethics. All data used in this study are publicly available and anonymized where necessary. No human subjects were directly involved. Potential biases in the datasets have been considered and mitigated. There are no conflicts of interest or sponsorship issues associated with this work.

Reproducibility Statement

To facilitate reproducibility, all code, model checkpoints, and datasets used in this paper are publicly available at `https://anonymous.4open.science/r/LONG1k-32B`. Detailed descriptions of the algorithms, hyperparameters, and data processing steps are provided in the paper and its appendix.

Acknowledgments

Acknowledgement: This research was supported by the Major Project of the National Social Science Foundation of China (Grant No. 24ATQ009).

References

Marthe Ballon, Andres Algaba, and Vincent Ginis. The relationship between reasoning and performance in large language models – o3 (mini) thinks harder, not longer. *arXiv preprint arXiv:2502.15631*, 2025.

DeepSeek-AI Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, Huazuo Gao, Kaige Gao, Wenjun Gao, Ruiqi Ge, Kang Guan, Daya Guo, Jianzhong Guo, Guangbo Hao, Zhewen Hao, Ying He, Wen-Hui Hu, Panpan Huang, Erhang Li, Guowei Li, Jiashi Li, Yao Li, Y. K. Li, Wenfeng Liang, Fangyun Lin, Aixin Liu, Bo Liu (Benjamin Liu), Wen Liu, Xiaodong Liu, Xin Liu, Yiyuan Liu, Haoyu Lu, Shanghao Lu, Fuli Luo, Shirong Ma, Xiaotao Nie, Tian Pei, Yishi Piao, Junjie Qiu, Hui Qu, Tongzheng Ren, Zehui Ren, Chong Ruan, Zhangli Sha, Zhihong Shao, Jun-Mei Song, Xuecheng Su, Jingxiang Sun, Yaofeng Sun, Min Tang, Bing-Li Wang, Peiyi Wang, Shiyu Wang, Yaohui Wang, Yongji Wang, Tong Wu, Yu Wu, Xin Xie, Zhenda Xie, Ziwei Xie, Yi Xiong, Hanwei Xu, Ronald X Xu, Yanhong Xu, Dejian Yang, Yu mei You, Shuiping Yu, Xin yuan Yu, Bo Zhang, Haowei Zhang, Lecong Zhang, Liyue Zhang, Mingchuan Zhang, Minghu Zhang, Wentao Zhang, Yichao Zhang, Chenggang Zhao, Yao Zhao, Shangyan Zhou, Shunfeng Zhou, Qihao Zhu, and Yuheng Zou. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Ma teusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Yuyang Ding, Xinyu Shi, Xiaobo Liang, Juntao Li, Qiaoming Zhu, and Min Zhang. Unleashing reasoning capability of llms via scalable question synthesis from scratch. *arXiv preprint arXiv:2410.18693*, 2024.

Ting Fu, Jiawei Gu, Yafu Li, Xiaoye Qu, and Yu Cheng. Scaling reasoning, losing control: Evaluating instruction following in large reasoning models. *ArXiv*, abs/2505.14810, 2025.

Tianyu Gao, Alexander Wettig, Howard Yen, and Danqi Chen. How to train long-context language models (effectively). *arXiv preprint arXiv:2410.02660*, 2024.

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.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Xiaodong Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and L. Sifre. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenyue Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. The impact of reasoning step length on large language models. *arXiv preprint arXiv:2401.04925*, 2024.

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

Koray Kavukcuoglu. Gemini 2.0 is now available to everyone, 2025. URL https://blog.google/technology/google-deepmind/gemini-model-updates-february-2025/. Accessed: 2025-03-06.

Dacheng Li, Shiyi Cao, Chengkun Cao, Xiuyu Li, Shangyin Tan, Kurt Keutzer, Jiarong Xing, Joseph Gonzalez, and Ion Stoica. S*: Test time scaling for code generation. *arXiv preprint arXiv:2502.14382*, 2025a.

Dacheng Li, Shiyi Cao, Tyler Griggs, Shu Liu, Xiangxi Mo, Eric Tang, Sumanth Hegde, Kourosh Hakhamaneshi, Shishir G. Patil, Matei Zaharia, Joseph Gonzalez, and Ion Stoica. Llms can easily learn to reason from demonstrations structure, not content, is what matters! *arXiv preprint arXiv:2502.07374*, 2025b.

Zhongzhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhijiang Guo, Le Song, and Cheng-Lin Liu. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025c.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harrison 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.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jian-Guang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*, 2023.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2, 2025. Notion Blog.

Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, and Jiahui Wen. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *arXiv preprint arXiv:2412.09413*, 2024.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Fei-Fei Li, Hanna Hajishirzi, Luke S. Zettlemoyer, Percy Liang, Emmanuel J. Candes, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Mathematical Association of America. Aime, February 2024. URL https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions/.

OpenAI. Learning to reason with llms, 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Jun-Mei Song, Mingchuan Zhang, Y. K. Li, Yu Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpaca: a strong, replicable instruction-following model; 2023. *URL https://crfm. stanford. edu/2023/03/13/alpaca. html*, 2023.

NovaSky Team. Sky-t1: Train your own o1 preview model within $450, 2025a.

OpenThoughts Team. Open Thoughts. https://open-thoughts.ai, January 2025b.

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025c. URL https://qwenlm.github.io/blog/qwq-32b/.

Jiaan Wang, Fandong Meng, Yunlong Liang, and Jie Zhou. Drt: Deep reasoning translation via long chain-of-thought. *arXiv preprint arXiv:2412.17498*, 2024a.

Shuhe Wang, Guoyin Wang, Jiwei Li, Eduard H. Hovy, and Chen Guo. Packing analysis: Packing is more appropriate for large models or datasets in supervised fine-tuning. *ArXiv*, abs/2410.08081, 2024b.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Yuyang Wu, Yifei Wang, Tianqi Du, Stefanie Jegelka, and Yisen Wang. When more is less: Understanding chain-of-thought length in llms. *arXiv preprint arXiv:2502.07266*, 2025.

Khurram Yamin, Shantanu Gupta, Gaurav R Ghosal, Zachary C Lipton, and Bryan Wilder. Failure modes of llms for causal reasoning on narratives. *arXiv preprint arXiv:2410.23884*, 2024.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin 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, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, Shanghaoran Quan, and Zekun Wang. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

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.

A PRELIMINARIES

A.1 DATASETS

Throughout this study, we rely on two primary datasets, which serve as the foundation for both our empirical investigations and the construction of the Long1K dataset. These datasets are selected for their high quality and alignment with our experimental goals, particularly because both are distilled from DeepSeek-R1, enabling fair comparisons with recent strong baselines.

- **Openthoughts-114k** Team (2025b): A large-scale dataset compiled from nine sources, covering four domains: Code, Math, Science, and Puzzle. It contains a total of 114,000 problems with corresponding reasoning traces and answers.
- **S1K-1.1** Muennighoff et al. (2025): A curated dataset consisting of 1,000 high-quality and diverse problems, selected from an initial pool of 59,029 problems across 16 sources. The selection process emphasizes quality, difficulty, and coverage diversity.

We evaluate our models on the following benchmarks, chosen for their reasoning-intensive nature and their frequent use in prior work:

- **AIME2024/2025** of America (2024): Two test sets each containing 30 math problems derived from the American Invitational Mathematics Examination, known for requiring deep analytical reasoning and multi-step problem solving.
- **MATH500** Hendrycks et al. (2021): A benchmark of 500 math problems covering a wide range of difficulty levels and mathematical domains, widely adopted for evaluating mathematical reasoning in language models.
- **GPQA Diamond** Rein et al. (2023): A benchmark containing 198 doctoral-level multiple-choice questions in physics, chemistry, and biology, designed to test deep scientific reasoning and factual understanding.

A.2 BASE MODELS

For most experiments, we use Qwen2.5-32B-Instruct Yang et al. (2024) as our base model. This choice is motivated by its proven effectiveness in training high-performance reasoning models, as demonstrated by recent state-of-the-art systems such as DeepSeek-R1-Distill-Qwen-32B Guo et al. (2025), S1.1-32B Li et al. (2025a), and LIMO Ye et al. (2025). Using the same model also facilitates fair comparisons across different training strategies.

We also use Qwen2.5-7B-Instruct Yang et al. (2024) in some complementary experiments, particularly when comparing smaller-scale setups. However, due to the limited availability of long reasoning samples, the 7B model is less stable under low-resource conditions, and most key results are obtained using the 32B variant.

A.3 TRAINING AND EVALUATION

All experiments are conducted using 8 NVIDIA A800-SXM4-80GB GPUs. Unless otherwise specified, we use 3 training epochs with a total batch size of 16. The learning rate is set to `2e-4`, and we adopt cosine learning rate scheduling. All models are trained using full-parameter fine-tuning.

For evaluation, we set temperature to 0.9 and top_p to 0.7. Accuracy is calculated based on the correctness of the final answer. We use Qwen2.5-72B-Instruct Yang et al. (2024) as an external verifier to compare model outputs against ground-truth answers. All reported scores are averaged over five independent runs to ensure stability.

B Extracting Mathematical Knowledge Points and Synthesizing Questions

We use the following prompt to extract knowledge points from mathematical problems. These knowledge points are then used to combine easy and difficult questions, with the prompts used for synthesis also shown here. To ensure fairness in the experiment, each question is synthesized using 4 knowledge points.

Extract Mathematical Knowledge Points Prompt

You will be given a problem, its corresponding solution, and a difficulty rating called `qwen_difficulty_parsed`, which ranges from 1 to 7. Your task is to extract at most two self-contained key points (knowledge points) from the problem and its solution. Each key point must be independent and descriptive, enabling the reader to fully understand the concept without additional context. Follow these rules:

1. Key Point Limit:
   - If `qwen_difficulty_parsed` ≤ 3, extract only one key point (major key point).
   - If `qwen_difficulty_parsed` ≥ 4, extract up to two key points (major and minor key points).
   - Keep the key points concise but meaningful. Avoid unnecessary splits or dependencies.

2. Structure: For each key point, include the following fields:
   - name: A concise title for the key point.
   - description: A clear, self-contained explanation of the concept, including its key ideas, principles, or definitions.
   - example: A minimal example illustrating the key point, either from the given problem or a simpler case.
   - assessment_scenarios: Provide two scenarios where this key point can be applied. The scenarios should show how the knowledge can be used in different contexts or types of problems.

3. Independence: Key points should not reference or depend on each other. Each key point should be understandable on its own.

4. Output Format: Your response must be a JSON array, where each object corresponds to a key point. Example structure:

```json
[
  {
    "name": "Key Point Name",
    "description": "Self-contained explanation of the key point.",
    "example": "A minimal example demonstrating the concept.",
    "assessment_scenarios": [
      "Scenario A: A description of where this key point might be tested or applied.",
      "Scenario B: Another context where this knowledge is relevant."
    ]
  },
  ...
]
```

Below are some examples:

Figure 4: This is the prompt for extracting mathematical knowledge points.

Examples of Extracting Mathematical Knowledge Points

problem: In triangle ABC, AB = 10 and AC = 17. Let D be the foot of the perpendicular from A to BC. If BD:CD = 2:5, then find AD.
solution: To solve for AD in triangle ABC where AB = 10, AC = 17, and BD:CD = 2:5, we follow these steps:
1. Let h = AD. We aim to find the value of h.
2. Apply Pythagoras' theorem in right triangle ABD:
  BD^2 = AB^2 - AD^2 = 10^2 - h^2 = 100 - h^2.
3. Apply Pythagoras' theorem in right triangle ACD:
  CD^2 = AC^2 - AD^2 = 17^2 - h^2 = 289 - h^2.
4. Given the ratio BD:CD = 2:5, we can express this as a ratio of squares:
  BD^2 : CD^2 = 4 : 25.
5. Set up the equation based on the ratio of squares and the expressions for BD^2 and CD^2:
  (100 - h^2) / (289 - h^2) = 4 / 25.
6. Solve for h:
  Cross-multiply to get 25(100 - h^2) = 4(289 - h^2).
  Simplify to find the value of h.
  Solving the equation, we find h = 8.

Therefore, the length of AD is 8.
qwen_difficulty_parsed: 4
extracted:
[
 {
  "name": "Pythagorean Theorem in Triangles",
  "description": "In a right triangle, the square of the hypotenuse is equal to the sum of the squares of the other two sides. This theorem is widely used for calculating unknown side lengths.",
  "example": "For a triangle with legs 3 and 4, the hypotenuse is sqrt(3^2 + 4^2) = 5.",
  "assessment_scenarios": [
   "Scenario A: Solve for the missing side of a triangle when given the hypotenuse and one leg length.",
   "Scenario B: Use the theorem to compute the diagonal of a rectangle in coordinate geometry."
  ]
 },
 {
  "name": "Using Ratios to Solve Geometric Problems",
  "description": "Ratios describe how a line segment is divided or the relationship between two geometric quantities. These can be converted into equations to solve for unknowns.",
  "example": "If a line segment is divided in the ratio 2:3, you can assign lengths 2k and 3k to the segments and solve for k.",
  "assessment_scenarios": [
   "Scenario A: Divide a triangle's base into specific ratios and solve for segment lengths.",
   "Scenario B: Apply ratios in geometric optimization problems, such as determining centroid locations."
  ]
 }
]

Figure 5: This is an example of extracting mathematical knowledge points.

Easy (Composite) Problem Prompt

Objective
Create a comprehensive mathematics problem that integrates multiple concepts. The problem should cover most of the provided concepts.

Problem Design Requirements
1. Multiple, Self-Contained Question:
  - The problem can consist of one or more questions, in order to cover more concepts.
  - The solver should have to discover the correct approach rather than being guided step-by-step.

2. Deep Integration of Concepts:
  - The concepts used should interact with each other naturally rather than being isolated computations.
  - Avoid structuring the problem as “solve this, then plug it into that” unless the connection is hidden.

3. Non-Trivial Solution Path:
  - The solution method should not be immediately obvious.
  - Solvers should need to deduce missing pieces rather than simply computing sequentially.

4. Bottleneck for Logical Depth:
  - Introduce a natural "bottleneck", a point where solvers must pause and think deeply before progressing.
  - Avoid problems where the difficulty is just “more steps” instead of requiring deeper reasoning.

5. Hidden Constraints and Self-Consistency:
  - Ensure that different problem components restrict each other in a way that forces unique solutions.
  - Let one part of the problem naturally control the form of another, creating a tight logical structure.

6. Use an Inverse Design Approach:
  - Instead of stacking steps forward, start from a strong final constraint and work backward to construct the problem.
  - This ensures the problem is coherent and elegant, rather than just a sequence of disconnected calculations.

7. Increasing Difficulty for Solution:
  - Avoid to add hints in the questions, e.g., use XX phenomenon to solve XXX. Instead, e.g., solve XXX.

Output Format
- Problem Statement: A comprehensive, concise, and engaging problem that may consist of multiple questions.
- Solution: A logically structured breakdown of how each constraint leads to the answer, revealing the hidden relationships.

Below are the concepts for you to create the problem. Note that the examples and assessment_scenarios are used to help you to understand the concept. You are not neccessarily to be limited by these examples or assessment_scenarios.

Figure 6: This is a prompt for Easy (Composite) Problem based on multiple mathematical concepts.

Difficult (Composite) Problem Prompt

Objective
Create a highly challenging mathematics problem that integrates multiple concepts in a deeply interconnected and non-trivial way. The problem should increase logical complexity rather than just adding steps, forcing solvers to uncover hidden structures rather than following a direct, linear approach.

Problem Design Requirements
1. Single, Self-Contained Question:
  - The problem should be one concise question, without breaking it into explicit sub-steps.
  - The solver should have to discover the correct approach rather than being guided step-by-step.

2. Deep Integration of Concepts:
  - The concepts used should interact with each other naturally rather than being isolated computations.
  - Avoid structuring the problem as "solve this, then plug it into that" unless the connection is hidden.

3. Non-Trivial Solution Path:
  - The solution method should not be immediately obvious.
  - Solvers should need to deduce missing pieces rather than simply computing sequentially.

4. Bottleneck for Logical Depth:
  - Introduce a natural "bottleneck", a point where solvers must pause and think deeply before progressing.
  - Avoid problems where the difficulty is just "more steps" instead of requiring deeper reasoning.

5. Hidden Constraints and Self-Consistency:
  - Ensure that different problem components restrict each other in a way that forces unique solutions.
  - Let one part of the problem naturally control the form of another, creating a tight logical structure.

6. Use an Inverse Design Approach:
  - Instead of stacking steps forward, start from a strong final constraint and work backward to construct the problem.
  - This ensures the problem is coherent and elegant, rather than just a sequence of disconnected calculations.

Hints for Constructing the Problem
- Hide Explicit Instructions:
  - Instead of saying "solve for \(x \), then compute squares," state a condition that forces the solver to discover how these are linked.

- Embed Constraints Subtly:
  - Example: Instead of "Factor \(x^2 + bx + c \)," give an indirect condition on the roots \(p, q \), such as a geometric or counting property.

- Interweave Concepts Effectively:
  - Example: Connect a combinatorial counting problem to a logarithmic constraint by requiring a disguised recurrence relationship.

- Introduce an Unexpected Twist:
  - The solver should have a "Eureka!" moment—a point where they realize how seemingly separate ideas actually connect.

Output Format
- Problem Statement: A single, concise, and engaging question that does not explicitly state the steps.
- Solution: A logically structured breakdown of how each constraint leads to the answer, revealing the hidden relationships.

Below are the concepts for you to create the problem. Note that the examples and assessment_scenarios are used to help you to understand the concept. You are not neccessarily to be limited by these examples or assessment_scenarios.

Figure 7: This is a prompt for Difficult (Composite) Problem based on multiple mathematical concepts.

C EXAMPLES OF FOUR DIFFERENT DATASETS

This section provides examples of four different datasets: Easy (Composite), Difficult (Composite), Easy (Double), and Difficult (Single), and reports the average difficulty of questions and pass@5 in each dataset.

We employed two methods to assess difficulty. First, we used the Qwen2.5-72B-Instruct model to directly evaluate the dataset, adopting the established AOPS (Art of Problem Solving) difficulty scale (1-10) with LLM assessment, following Sky-T1 Team (2025a)). This approach does not involve simple binary classification but rather a structured evaluation process where the LLM provides extensive chain-of-thought reasoning before assigning difficulty scores, categorizing problems into human-interpretable levels. Second, we utilized the Qwen2.5-72B-Instruct model to generate five solutions for each of the four datasets and calculated the pass@5 metric to reflect the difficulty of each dataset. Through these two difficulty evaluation metrics, we measured the difficulty levels across the four datasets. The consistency between the two metrics validates the effectiveness of difficulty differentiation across different datasets.

Table 6: Examples of four different datasets: Easy (Composite), Difficult (Composite), Easy (Double), and Difficult (Single).

Dataset	**Avg. Difficulty**	**pass@5**	**Example**
Easy (Composite)	3.78	85.7%	Consider a convex pentagon with side lengths x, y, z, w, v, where $xyzwv = 1$. The pentagon can be divided into three triangles by drawing perpendiculars from one vertex. The areas of these triangles are proportional to the side lengths adjacent to the vertex. **1.** Given that $x^2 \leq 4$, find the maximum possible area of the pentagon. **2.** How many distinct sets of positive integer side lengths satisfy the given conditions?
Difficult(Composite)	4.44	75.5%	Determine the range of possible values for the angle θ such that the rotation matrix $R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$ satisfies the condition that the quadratic equation has no real roots, given the dimensions of a rectangular box whose length, width, and height are the arithmetic mean, geometric mean, and harmonic mean of two distinct positive real numbers x and y, respectively.
Easy (Double)	4.57	70.2%	**Question 1** is "48 blacksmiths need to shoe 60 horses. What is the minimum time they will spend on the job if each blacksmith takes 5 minutes per horseshoe?". **Question 2** is "If 25,197,624 hot dogs are packaged in sets of 4, how many will be left over?"
Difficult (Single)	5.01	56.4%	**Question 1** is "A straight ladder AB of mass $m = 1$ kg is positioned almost vertically such that point B is in contact with the ground with a coefficient of friction $\mu = 0.15$. It is given an infinitesimal kick at the point A so that the ladder begins rotating about point B. Find the value ϕ_m of angle ϕ of the ladder with the vertical at which the lower end B starts slipping on the ground."

D SCALING REASONING LENGTH OF TRAINING DATA

Across both the MATH500 and GPQA Diamond benchmarks, performance increases consistently as the reasoning length of the training data grows. Interestingly, the relationship follows a near-linear trend when plotted on a log-log scale, suggesting power-law behavior. To quantify this, we fit the following power-law model:

$$y = Ax^B,$$

where x is the average number of reasoning tokens per problem (N), and $\hat{y}$ is the estimated precision. We performed an ordinary least squares (OLS) regression of $\ln y$ on $\ln x$ after log-transforming the data. The fitted equations are:

$$\hat{y}_{\text{MATH500}}(N) = 0.6676\ N^{0.03720}, \qquad R^2 = 0.9711, \quad p = 3.17 \times 10^{-4}, \tag{1}$$

$$\hat{y}_{\text{GPQA}}(N) = 0.4095\ N^{0.04613}, \qquad R^2 = 0.9694, \quad p = 3.54 \times 10^{-4}. \tag{2}$$

Both fits show highly significant scaling behavior ($p < 0.001$), confirming a strong correlation between reasoning length and performance. These findings suggest that, across both mathematical and scientific reasoning tasks, exponential increases in training-sequence length can produce near-linear performance gains. This highlights the effectiveness and scalability of training with longer reasoning traces.

E PROMPT FOR SYNTHESIZING LONG1K

This section provides the complete prompts and output format used for synthetic data. Here, PROBLEM-i, THINK-i, and SOLUTION-i are all derived from a single sampled problem from a reasoning dataset (e.g., Openthoughts-114k). The underlined texts are templates used to link problems, thoughts, and solutions. We ask GPT-4o to produce multiple paraphrases of these templates for diversity.

Table 7: Prompts and output format for synthetic data.

System Prompt	<|im_start|>system\n A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within <|begin_of_thought|><|end_of_thought|>and <|begin_of_solution|><|end_of_solution|>tags, respectively, i.e., <|begin_of_thought|>reasoning process here <|end_of_thought|><|begin_of_solution|>answer here <|end_of_solution|>. <|im_end|>\n
User Prompt	<|im_start|>user\n I need help with the following problems. The first one is **PROBLEM-1**, the second one is **PROBLEM-2** ... , and the final one is **PROBLEM-N**. <|im_end|>\n
Output	<|im_start|>system\n <|begin_of_thought|>\n\n I will handle these problems one by one. I will start with the first problem. **THINK-1**.\n Now I will turn to the second problem. **THINK-2**. ... OK, now is the last problem. **THINK-N** <|end_of_thought|>\n\n <|begin_of_solution|>\n\n The solution for the first problem is as belows. **SOLUTION-1**.\n The solution for the second problem is as belows. **SOLUTION-2**. ... The solution for the last problem is as belows. **SOLUTION-N**.<|end_of_solution|> <|im_end|>

F BASELINE FOR MAIN RESULTS

We fine-tuned Qwen2.5-32B-Instruct on the Long1K dataset to obtain our final model, **Long1K-32B**, with a maximum output length of 32k tokens.

To evaluate its performance, Long1K-32B was compared against several strong baselines on standard reasoning benchmarks. All models are fine-tuned on Qwen2.5-32B-Instruct for consistency. The baselines include:

- **S1-32B** and **S1.1-32B**: Both models are trained on 1,000 problems. S1-32B uses solutions distilled from Gemini, while S1.1-32B uses those distilled from DeepSeek-R1.
- **LIMO** Ye et al. (2025): A model trained on 800 high-quality, diverse, and challenging problems. It emphasizes careful data curation over scale.
- **OpenThinker-32B** Team (2025b): A model trained on the full 114k examples of the Openthoughts-114k dataset.
- **DeepSeek-R1-Distill-Qwen-32B** Guo et al. (2025): A large-scale model trained on over 800k reasoning examples. Note that the training data for this model has not been publicly released.

All baselines, except S1-32B, are trained on reasoning traces distilled using DeepSeek-R1, and all are fine-tuned on the same base model (Qwen2.5-32B-Instruct). This consistent setup ensures a fair comparison, allowing us to isolate and assess the benefit brought by *longer reasoning traces* in the Long1K dataset.

G IMPROVED LONG-RANGE INSTRUCTION FOLLOWING

We compared the performance of Long1K-32B and S1.1-32B models across all test sets of the MathIF dataset. The MathIF dataset is specifically designed to systematically evaluate the instruction-following capability of large language models in mathematical reasoning tasks. It introduces three progressively complex constraint levels (Single, Double, and Triple) that incrementally increase demands on model reasoning and compliance. The evaluation framework comprises three metrics: HAcc (which reflects both mathematical and instruction-following abilities), SAcc (which solely reflects instruction-following ability), and Correctness (a baseline metric that only considers the correctness of the final answer)

Table 8: Instruction-following performance across different response lengths.

Test	**S1.1-32B**			**Long1K-32B**		
	HAcc(%)	SAcc(%)	Correctness(%)	HAcc(%)	SAcc(%)	Correctness(%)
GSM8K-single	46.7	46.7	80.0	80.0	80.0	90.0
GSM8K-double	6.7	38.3	86.7	43.3	70.0	90.0
GSM8K-triple	16.7	48.8	86.7	36.7	73.3	86.7
MATH500-single	40.0	40.0	93.3	66.7	66.7	93.3
MATH500-double	6.7	35.0	80.0	30.0	70.0	76.7
MATH500-triple	3.3	43.3	86.7	33.3	70.0	86.7
MINERVA-single	33.3	36.7	36.7	36.7	76.7	76.7
MINERVA-double	13.3	40.0	36.7	30.0	58.3	33.3
MINERVA-triple	23.3	63.3	43.3	30.0	72.2	30.0
OLYMPIAD-single	26.7	26.7	53.3	50.0	60.0	60.0
OLYMPIAD-double	16.7	41.7	43.3	26.7	51.7	43.3
OLYMPIAD-triple	13.3	54.4	50.0	33.3	71.1	53.3
AIME-single	35.0	40.0	40.0	35.0	70.0	70.0
AIME-double	5.0	35.0	35.0	10.0	40.0	45.0
AIME-triple	5.0	40.0	25.0	10.0	53.3	25.0

H LIMITATIONS

While our study highlights the importance of reasoning length, several limitations remain.

First, due to computational constraints, we only scale training up to 32k tokens. As observed in our instruction-following experiments, the instruction adherence rate drops to around 81% for the longest inference group, suggesting that further gains could be achieved with even longer training sequences.

Second, our experiments are limited to knowledge distillation Hinton et al. (2015). We have not yet explored applying long reasoning chains directly in reinforcement learning settings (e.g., RLHF) Luo et al. (2025) to enhance native reasoning abilities.

Third, our evaluation focuses on mathematical and scientific reasoning tasks. Whether these findings generalize to broader open-domain or commonsense reasoning remains to be verified.

I STATEMENT OF LLM USAGE

Large language models were used to aid or polish writing and to assist with retrieval and discovery of related work. All technical content, experimental design, and data analysis decisions were made independently by the authors, and the final manuscript was reviewed and edited by the authors.