

Long Chain-of-Thought Fine-tuning via Understanding-to-Reasoning Transition

Anonymous ACL submission

Abstract

Reasoning models have demonstrated remarkable performance on complex tasks by generating long reasoning traces prior to producing final answers. However, previous research on long-context scaling in language models has generally focused on managing lengthy input prompts instead of producing long outputs. To leverage the strong long context understanding abilities of current models, we introduce Understanding-to-Reasoning Transition (URT) fine-tuning, a sequence-level curriculum learning framework that gradually shifts a model’s focus from interpreting long chain-of-thoughts to generating them. By incorporating partial reasoning steps in the input context, URT naturally exposes the model to diverse prompt lengths during training, preserving its performance on long-context comprehension while developing advanced reasoning capabilities. Experiments on rigorous reasoning benchmarks, including AIME24 and GPQA Diamond, reveal that our approach surpasses standard fine-tuning by over 10%, while maintaining robust performance on the understanding tasks in RULER.

1 Introduction

Enhancing the long-context capabilities of large language models (LLMs) (Anthropic, 2023; Touvron et al., 2023; Reid et al., 2024) has become both immensely popular and critically important. In recent years, researchers have primarily focused on improving the long-context comprehension abilities of LLMs (Xiong et al., 2023; Zhu et al., 2023; Peng et al., 2023; Gao et al., 2024b), achieving substantial progress on tasks such as summarization and question answering that emphasize the effective utilization and understanding of long inputs. However, with the emergence of reasoning models (OpenAI, 2024; DeepSeek-AI et al., 2025), the focus has noticeably shifted; optimization in the long-context setting is now aimed at generat-

ing long outputs, typically in the form of extensive chain-of-thoughts (CoTs) (Wei et al., 2023).

However, it remains unclear (1) whether current open-source reasoning models that are trained with short prompts can preserve their long-context comprehension capabilities; and (2) whether their abilities to handle long prompts can be effectively leveraged to enhance the learning of long CoTs. To address this gap, our study begins with a preliminary evaluation (§3.1) examining whether popular base LLMs and their reasoning variants can effectively utilize information from lengthy inputs. First, to assess the models’ general long-context capabilities, we adopt the existing benchmark RULER (Hsieh et al., 2024), which measures performance on standardized long-context tasks. In addition to the general domain, we introduce **CoT-U** (Chain-of-Thought Understanding), a new understanding task that evaluates models’ ability to process and reason over lengthy inputs which are human-verified reasoning paths. As illustrated in Figure 1, each input combines a problem statement with its corresponding CoT rationale, testing how effectively models leverage structured reasoning paths. Our evaluation leads to two key observations: (1) Reasoning models (trained predominantly on math problem with limited prompt-length variety) lag behind base models in both CoT-U and general long context tasks, and (2) incorporating partial reasoning steps (tokens) into the input significantly mitigates the difficulty of learning long reasoning chains.

Building on these findings, we propose a new framework for fine-tuning LLMs on challenging chain-of-thought data, termed *Understanding-to-Reasoning Transition* (URT) fine-tuning. In this context, *understanding* refers to the model’s ability to process long-form CoT inputs and assimilate them into accurate final answers. While *reasoning* in our approach involves generating the reasoning steps rather than merely interpreting them. Recent studies, e.g., Sky-T1 and S1 (DeepSeek-AI

et al., 2025; NovaSky Team, 2025; Ye et al., 2025; Muennighoff et al., 2025), have demonstrated that fine-tuning on long CoT data offers a straightforward and cost-effective approach to enabling models to learn extended CoTs. In contrast to these works that finetuned models on complete reasoning chains, URT models the learning process as a gradual transition from understanding to generation. Specifically, URT splits a long reasoning trace into two segments—incorporating part of the trace into the input while requiring the generation of the remaining steps. This means that during training, the input prompt is gradually shortened while the expected output becomes longer. Our approach offers two key benefits. First, it **enhances reasoning** by employing a sequence-level curriculum learning process that bridges the gap between long chain-of-thought understanding and generation by starting with an easier understanding task. Second, it **improves comprehension** by exposing the model to diverse prompt lengths during fine-tuning, thereby avoiding overfitting on short-prompt inputs.

We evaluate our URT fine-tuning method on several challenging reasoning benchmarks, including AIME24 (MAA, 2024), GPQA (Rein et al., 2023), and MATH500 (Hendrycks et al., 2021a), as well as a long-context benchmark, RULER (Hsieh et al., 2024). Our experiments cover various model sizes (8B, 14B, and 32B), and in every case, URT fine-tuning consistently delivers improvements. Notably, our 32B model achieves an improvement of over 10% compared to standard SFT methods. The overall performance of our model is even comparable to that of the teacher models R1-Preview and QwQ-Preview, demonstrating its impressive capacity to acquire and generalize knowledge. In addition, for long-context understanding tasks, our model achieves robust performance on RULER, outperforming QwQ-32B-preview by over 12 points.

2 Related Work

Long-Context Understanding Language models have recently increased their context lengths from 2K to over 128K tokens (Chen et al., 2023; Han et al., 2024; Gao et al., 2024b; Lin et al., 2025). Despite this growth, evaluation and training datasets have mainly concentrated on comprehending long prompts (Li et al., 2024; Jiang et al., 2024). Benchmarks like Scrolls (Shaham et al., 2022) and LEval (An et al., 2023) assess long-

context capabilities through tasks such as question answering (Kočíský et al., 2018; Dasigi et al., 2021), summarization (Zhong et al., 2021; Chen et al., 2022), and synthetic challenges like Needle-in-a-Haystack (gkamradt, 2023; Laban et al., 2024). For training, efforts focus on curating high-quality long-context data from real-world QA datasets, typically following a long-prompt, short-response format (Zhao et al., 2024a; An et al., 2024; Koluguri et al., 2024; Liu et al., 2024; Fu et al., 2024; Wu et al., 2024; Gao et al., 2024a).

Long Chain-of-Thought Reasoning As LLMs’ context lengths expand, research also focuses on generating extended content, such as story writing (Mikhaylovskiy, 2023; Bai et al., 2024). With the release of o1 (OpenAI, 2024), generating long, coherent chains of thought (CoTs) in model responses has become a critical research area (Zhang et al., 2024a; Latif et al., 2024). Enhancements in long CoT generation include Monte Carlo Tree Search (Yao et al., 2023; Zhou et al., 2024; Zhang et al., 2024c), large-scale reinforcement learning (DeepSeek-AI et al., 2025; Pan et al., 2025), and supervised fine-tuning on CoT datasets (Team, 2025; LI et al., 2024). We use supervised fine-tuning for its scalability and strong reasoning performance (Huang et al., 2025; Zhang and Chen, 2024; Muennighoff et al., 2025). Unlike data-centric approaches focused on data diversity and quality (Ye et al., 2025; Min et al., 2024; NovaSky Team, 2025), our work optimizes the training process with a new sequence-level finetuning framework without complex data engineering.

3 Method

This section comprises two primary components: (1) a preliminary evaluation of the long-context capabilities of current models, and (2) our URT-based fine-tuning framework.

3.1 Long Context Understanding Capabilities of Current LLMs

In this section, we describe the details of the long-context understanding evaluation process of current base models and reasoning models.

Evaluation Setup Base models have demonstrated remarkable progress in long-context understanding, as evidenced by their performance on various benchmarks (Zhang et al., 2024b; Song et al., 2025; Bai et al., 2025). However, their ability

to comprehend complex chain-of-thought (CoT) reasoning and the effectiveness of reasoning models for processing long inputs remain largely unexplored. In general long-context ability tests, we utilize RULER (Hsieh et al., 2024) to evaluate these models. We note that there is currently no understanding task featuring long CoT inputs. In this work, we introduce **CoT-U**, a new test dataset designed to assess a model’s ability to retrieve and aggregate information from long chain-of-thought reasoning traces. Given a problem statement accompanied by its multi-step reasoning process as input context, the model must aggregate the useful information to arrive at the correct final answer (see Figure 1).

The construction of CoT-U requires not only the problems but also their corresponding correct reasoning steps. Our experiments are conducted on a dataset comprising 180 math problems and their corresponding solutions, sourced from the Art of Problem Solving (AoPS) platform¹ and spanning the AIME competitions from 2019 to 2024. The AoPS platform provides Olympiad-level math problems accompanied by detailed, human-authored CoTs. In addition to these human-written solutions, we also develop a set of model-generated reasoning paths. Unlike the human solutions, the synthetic paths are enriched with more elaborate explanations and reflective steps, and thus longer and more challenging. Given that o1 (OpenAI, 2024) lacks explicit reasoning tokens, we employ the state-of-the-art open-source model QwQ-32B-Preview from Qwen Team (Bai et al., 2023) to generate these solutions.

From the initial set of 180 generated solutions, we apply rejection sampling to eliminate 56 instances—those with incorrect answers (identified via rule-based matching) and those with correct answers but insufficient or flawed reasoning (determined through human review). In CoT-U, we do not directly include the correct answer in the input. Instead, we remove any sentences that explicitly reveal the ground truth. The model is prompted with the incomplete reasoning process which have already demonstrate the key reasoning steps for deducing the answer. After comprehending this solution, the model only needs to perform simple reasoning to arrive at the final answer. All the data are then subject to manual review to ensure that the reasoning steps are logically sound and free from

```
<prompt>
A bug walks all day and sleeps all night. On the first
day, it starts at point O, facing east, and walks 5
units due east. Each subsequent day, it walks half as
far as the previous day in a direction that is 60°
counterclockwise ... Find m+n. So I have this problem
about a bug that walks different distances each day
and rotates 60 degrees counterclockwise every night.
It starts at point O, facing east, and walks 5 units
east on the first day. Then each subsequent day, it
walks half as far as the previous day in a new
direction that's 60 degrees counterclockwise from
where it was facing the previous night ... Then,  $OP^2 =$ 
 $x^2 + y^2 = 25 + (25 \cdot 3)/9 = 25 + 75/9 = 25 + 25/3 = (75$ 
 $+ 25)/3 = 100/3$ 
Therefore,  $m = 100$  and  $n = 3$ , so  $m + n = 103$ . wait, but
earlier I had a different ... (removed)

User: The provided solution has sufficient information
to derive the answer to the question. Please
understand the solution and obtain the final answer.
</prompt>
```

Figure 1: An illustrative example from our CoT-U test set, where we remove sentences that directly reveal the final answer and instead include an instruction prompting the model to comprehend the given CoTs.

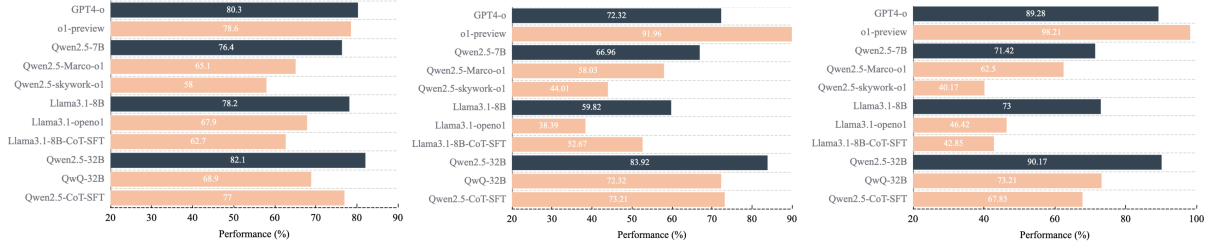
overt mistakes, resulting in a refined test set of 112 cases. Human-generated chains-of-thought average 872 tokens, while AI-generated chains are significantly longer at 16,133 tokens (approximately 18 times longer).

Findings We evaluate the long thought understanding ability of GPT-4o (OpenAI, 2023), Llama3.1-8B (Llama Team, 2024), Qwen2.5-7B/32B (Qwen et al., 2024), and their corresponding long reasoning models: o1-preview (OpenAI, 2024), Qwen2.5-marco-o1 (Zhao et al., 2024b), Qwen2.5-skywork-o1 (o1 Team, 2024), Llama3.1-open-o1², and QwQ-32B-preview on CoT-U. Results are presented in Figure 2 and Figure 3. Our results reveal the following insights:

- (1) All open-source reasoning models exhibit a significant performance degradation in long-context understanding tasks compared to their non-reasoning counterparts.
- (2) Without any hints or reasoning steps provided, open-source reasoning models achieve superior performance in solving mathematical problems. However, when supplied with CoTs generated by other AI systems or humans, their problem-solving capability paradoxically underperforms compared to non-reasoning models.
- (3) Model performance is closely associated with the length of the provided context. Figure 3 illustrates the continuous improvement in performance of Qwen2.5-32B-Instruct and Llama3.1-8B-Instruct as they are supplied with more reasoning steps. Specifically, Qwen2.5-32B-Instruct demonstrates a significant improvement

¹<https://artofproblemsolving.com/>

²<https://github.com/Open-Source-O1/Open-O1>



(a) Results on RULER(32K). (b) Results on CoT-U (human-written). (c) Results on CoT-U (AI-generated).

Figure 2: Long context understanding ability of current LLMs. Black bars: base models; orange bars: long reasoning models. Qwen2.5-CoT-SFT and Llama3.1-CoT-SFT is trained by us on the STILL2 dataset (Min et al., 2024).

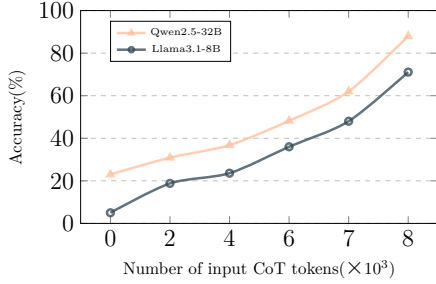


Figure 3: Model performance with varying numbers of provided CoT tokens.

of over 60% compared to directly prompting the model with the problem statement which means the accuracy of solving math problems can be controlled by the number of reasoning tokens.

3.2 Fine-Tuning on Long Thought data

The evaluation results highlights the need for advanced fine-tuning techniques to enhance the long CoT capabilities of reasoning models without compromising their long-context understanding capabilities. This section details our URT fine-tuning method.

Formulation Consider the training dataset defined as

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^N,$$

where $x^{(i)}$ is an input instance (e.g., a question or prompt) and $y^{(i)}$ is the corresponding long chain-of-thought response, which includes intermediate reasoning steps and the final answer.

The LLM, parameterized by θ , defines a conditional distribution over outputs as $p(y | x; \theta)$. For a single training instance (x, y) , the loss is defined by:

$$\mathcal{L}(\theta; x, y) = -\log p(y | x; \theta).$$

The overall objective for fine-tuning is to minimize

the total loss over the entire dataset:

$$\mathcal{L}(\theta) = \frac{1}{|D|} \sum_{(x^{(i)}, y^{(i)}) \in D} \mathcal{L}(\theta; x^{(i)}, y^{(i)}).$$

This formulation represents the traditional Supervised Fine-Tuning (SFT) method. However, in practice, when dealing with long reasoning chains, the response y is often highly complex and lengthy (Xu et al., 2025; Wang et al., 2024), posing significant challenges for effective model learning.

To overcome the challenges associated with long reasoning chains, we propose Understanding-to-Reasoning Transition Supervised Fine-Tuning (URT-SFT), which involves partitioning the response y into two distinct parts and embedding a segment of y within the input prompt. Specifically, the response y is divided as follows:

$$y = y_{1:t} \circ y_{t+1:T},$$

where: $y_{1:t} = \{y_1, y_2, \dots, y_t\}$, $y_{t+1:T} = \{y_{t+1}, y_{t+2}, \dots, y_T\}$, and \circ denotes the concatenation of sequences. Here, t represents the length of the segment from y that is integrated into the input prompt, which is a hyperparameter. The augmented input x' is then defined as:

$$x' = x \circ y_{1:t},$$

where x is the original input prompt augmented with the first t tokens of the response y .

Consequently, the loss function for a single training instance under URT-SFT is formulated as:

$$\mathcal{L}(\theta; x, y) = -\log p(y_{t+1:T} | x'; \theta).$$

This splitting strategy enables the model to handle long reasoning chains more effectively by reducing the complexity of the output. Furthermore, in traditional SFT, most inputs x are math problems (Min et al., 2024; NovaSky Team, 2025; LI

Table 1: Examples of different fine-tuning strategies on long reasoning data. During training, the prompt and response are separated by the special token ‘</prompt>’. The lengths of the prompt and response can be adjusted by modifying the placement of </prompt>. Instead of fine-tuning on the entire response, our method begins with an input format that is closer to long-context understanding and gradually transitions to training on the whole thought.

Fine-tuning on whole thought	<prompt> A bug walks all day and sleeps all night... Find $m+n$ </prompt> (no loss on prompt) So I have this problem about a bug that walks different distances each day and...(the whole thought has about 5,120 tokens) the final answer is 103
From long thought understanding to long thought generation	<prompt> A bug walks all day and sleeps all night... Find $m+n$ So I have this problem about a bug that walks different distances...(the input context contains 4,096 tokens) </prompt> Now, $P = 5/0.75 - i(\sqrt{3}/4)$. To divide by a complex number, multiply numerator and denominator by the conjugate of the denominator...(the output contains about 1024 tokens) the final answer is 103
	<prompt> A bug walks all day and sleeps all night... Find $m+n$ So I have this problem about a bug that walks different distances...(the input context contains 2,048 tokens) </prompt> First day: walk 5 units east, so position $Z_1 = 5 + 0i$. Then, rotate 60 degrees...(the output contains about 3072 tokens) the final answer is 103
	Fine-tuning on whole thought (the output contains about 5120 tokens)

et al., 2024), which are naturally short. However, after concatenation, x' becomes a long-context input. Exposing the model to a range of input lengths during fine-tuning demonstrably enhances the performance of long reasoning models on long-context tasks (§4.4).

An Example of Decomposing In this section, we demonstrate how we split a long thought, with t set to powers of 2, specifically {2,048, 4,096, 8,192}. The thought y consists of approximately 5,120 tokens, as illustrated in Table 1. We utilize a simple chat template where the user input is enclosed within <prompt>...</prompt>. Following previous work (Wang et al., 2023; Zheng et al., 2024), we do not calculate loss on the prompt.

In the top block of Table 1, the model is fine-tuned on the entire chain-of-thought y , where the prompt x contains only a brief problem description (e.g., “A bug walks all day and sleeps all night... Find $m + n$ ”), and the full reasoning (about 5,120 tokens) along with the final answer is produced in the response. The subsequent blocks illustrate our method of decomposing the long thought—transitioning from long thought understanding to long thought generation. Given that the input length of 5,120 tokens is below the 8,192 tokens threshold, we experiment with splitting points $t = 2,048$ and 4,096. In our implementation, we first split the text by $\backslash n \backslash n$ and then by length. This ensures that splitting points are at $\backslash n \backslash n$ positions, preventing tokenization errors. In the first row of the second block, the prompt includes a portion of the chain-of-thought (with the context containing $t = 4,096$ tokens), and the response provides the remaining reasoning steps (approximately 1,024

tokens) together with the final answer. In the second row, we increase the task difficulty by reducing the input context length to around $t = 2,048$ tokens, resulting in a longer response (about 3,072 tokens). Finally, the model is trained on the whole thought without any reasoning steps provided. The training details are in Section §4.1.

Difficulty Control As a sequence-level curriculum learning framework, URT is designed to mimic an easy-to-hard training progression. At the sequence level, we emphasize training examples with longer prompts—providing additional context or intermediate reasoning—paired with shorter responses. This design encourages the model to leverage extended prompts containing rich reasoning information while reducing the burden of generating lengthy responses. At the beginning of every epoch, the training data is shuffled. For identical math problems, training samples that incorporate more reasoning steps are prioritized. It means the model is initially exposed to instances where more intermediate steps help in reaching the final answer. This ordering facilitates a gradual increase in difficulty as training progresses. After each epoch, the model is evaluated on the training set. If it correctly derives a problem’s final answer, that problem is removed, allowing the model to concentrate on unsolved challenges.

4 Experiments

4.1 Training Details

In this section, we describe the data preparation, training process, and hyperparameter settings used in our experiments. Our focus is on the training

methodology rather than on curating or annotating data; therefore, we directly use publicly available, open-source dataset provided by STILL2 (Min et al., 2024) where the teacher models are QwQ-32B-preview (Qwen Team, 2024) and R1-lite-preview (DeepSeek Team, 2024).

Data Preparation We utilize the dataset provided in STILL2³ which comprises approximately 4.9K question and chain-of-thought pairs. The majority of the prompts are from the mathematics domain, with a smaller portion from code and science. In the original dataset, there are 4,900 prompts in total, with 2,429 having lengths exceeding 2,048 tokens, 1,071 exceeding 4,096 tokens, and 81 exceeding 8,192 tokens. We then create a decomposed dataset using the method illustrated in Table 1, consisting of 4,814 prompts with diverse input lengths (2,048~8,192).

Training The SFT baselines are trained on 4.9K prompts for 10 epochs, while the URT models are trained on 9.8K prompts for 5 epochs. Additionally, we exclude data that the model can already handle correctly. To validate the effectiveness of our approach, we perform experiments on three base models with sizes ranging from 8B to 32B: Llama3.1-8B-Instruct, Qwen2.5-14B-Instruct, and Qwen2.5-32B-Instruct. We do not modify the models’ tokenizers or chat templates, and we adopt a full-parameter fine-tuning setting. We use Llama Factory (Zheng et al., 2024) for fine-tuning and follow their default SFT hyperparameters. We use a batch size of 64, training in bfloat16 precision. The learning rate is set to 1×10^{-5} , with no warm-up, and it decays to 0 following a cosine schedule. We employ the AdamW optimizer (Loshchilov and Hutter, 2019) with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and a weight decay of 1×10^{-4} . Note that during training we compute the loss only on the reasoning traces from the response, and not on the input (which includes the question and any partial solution). All experiments are conducted on a single machine equipped with 8 NVIDIA A100 GPUs, and we set the maximum training sequence length to 16384. Our training pipeline is based on DeepSpeed ZeRO-3 (Aminabadi et al., 2022) and accelerated by Flash-Attention-2 (Dao, 2023). For the 32B model, CPU offload is utilized to mitigate GPU out-of-memory. In our experiments, the 8B and 14B models are

fine-tuned in under 4 hours, while the 32B model requires approximately 14 hours.

4.2 Evaluation Setup

In this section, we outline the benchmarks and baselines used to evaluate the reasoning capabilities of our models. Specifically, we compare URT with the traditional SFT on widely recognized reasoning tasks covering multiple domains as well as long-context benchmarks. For a detailed description of the benchmarks and baselines employed in this work, please refer to Section A.1.

4.3 Main Results

Long-Reasoning Results Table 2 presents a comprehensive comparison of various models evaluated on three benchmark datasets: MATH500, AIME24, and GPQA-Diamond. The results demonstrate that URT-SFT consistently outperforms the standard SFT approach across all evaluated models. For instance, applying URT-SFT to the Llama3.1-8B-Instruct model results in accuracy improvements of 17.6% on MATH500, 506.1% on AIME24, and 30.6% on GPQA-Diamond, greatly surpassing the SFT baseline. Similarly, URT-SFT outperforms SFT on larger base models like Qwen2.5-14B-Instruct and Qwen2.5-32B-Instruct.

Our proposed URT-SFT method leverages only 5K prompts to fine-tune models, yet achieves performance that rivals or exceeds that of our teacher models R1-preview and QwQ-32B-preview. Specifically, on the AIME24 dataset, our method achieves an accuracy of 56.7%, outperforming r1-preview’s 52.5% and QwQ-32B-preview’s 50.0%. Furthermore, in the GPQA-Diamond benchmark, our 32B model reaches an accuracy of 61.6, surpassing r1-preview by 3 points, demonstrating competitive performance despite the use of fewer prompts. When assessed against contemporary state-of-the-art open-source models, our URT-SFT-trained models also exhibit competitive performance across all benchmarks. On the AIME24 dataset, our approach aligns closely with S1-32B + BF, which is trained on data distilled from the stronger teacher model Gemini 2.0 Flash Thinking (Google Team, 2024). Our 32B model is able to outperform strong open-source baselines STILL2-32B and Sky-T1-32B, which employ similarly-performing teacher models. The results on the GPQA-Diamond show that our method clearly outperform all previous models when tested with a broader range of domains. We also demonstrate

³https://github.com/RUCAIBox/Slow_Thinking_with_LLMs

Table 2: Performance comparison of various models across three popular benchmarks to assess the long trace reasoning ability of LLMs. **Distilled From** indicates which model provides the training data. The **bold** fonts denote the best performance among our training variants, and we report the gain over the base model. The examples of the input and output for the three benchmarks are shown in Appendix Table 5.

Models	Distilled From	MATH500		AIME24		GPQA-Diamond	
		Acc (%)	Gain (%)	Acc (%)	Gain (%)	Acc (%)	Gain (%)
GPT-4o	N.A.	76.6	-	9.3	-	53.6	-
Claude 3.5 Sonnet	N.A.	78.3	-	16.0	-	65.0	-
o1	N.A.	94.8	-	74.4	-	77.3	-
o1-preview	N.A.	85.5	-	44.6	-	72.3	-
r1	N.A.	97.3	-	79.8	-	71.5	-
r1-preview	N.A.	91.6	-	52.5	-	58.5	-
QwQ-32B-preview	N.A.	90.6	-	50.0	-	65.2	-
Trained on open-source dataset							
Open-o1-Llama3.1-8B	Synthetic data	54.8	-	10.0	-	30.8	-
S1-32B + BF	Gemini Flash Think.	93.0	-	56.7	-	59.6	-
STILL2-32B	r1-preview & QwQ-preview	89.2	-	43.3	-	55.0	-
Sky-T1-32B	QwQ-preview	86.4	-	43.3	-	56.8	-
Llama3.1-8B-Instruct + SFT	r1-preview & QwQ-preview	48.4	-	3.3	-	23.2	-
+ URT-SFT		52.2	+7.9%	13.3	+303.0%	27.2	+17.2%
		57.0	+17.6%	20.0	+506.1%	30.3	+30.6%
Qwen2.5-14B-Instruct + SFT	r1-preview & QwQ-preview	78.6	-	13.3	-	43.4	-
+ URT-SFT		83.2	+6.4%	33.3	+150.4%	53.5	+23.3%
		86.0	+9.4%	40.0	+200.7%	60.6	+39.6%
Qwen2.5-32B-Instruct + SFT	r1-preview & QwQ-preview	80.0	-	13.3	-	43.4	-
+ URT-SFT		88.2	+11.5%	43.3	+225.5%	55.1	+26.9%
		90.4	+12.8%	56.7	+326.3%	61.6	+41.9%

that URT-SFT maintains an advantage over SFT in more tasks (Table 3).

Table 3: Performance across a broader range of tasks, including medicine, commonsense reasoning, code, and math. LCB stands for LiveCodeBench, and PA stands for Putnam-AXIOM.

Model	MedQA	BBH	LCB	PA
Qwen2.5-32B	72.0	50.8	40.8	17.7
+SFT	73.2	58.6	41.3	38.1
+URT-SFT	75.7	59.3	44.5	43.2

Long-Context Results As described in Section 3.1, the model’s ability to understand long reasoning traces deteriorates after extended CoT training, despite the task being a straightforward mathematical one. We hypothesize that this decline is due to the model not encountering long input prompts during training, which hampers its long-context understanding capabilities. To validate this hypothesis, we conducted evaluations using the RULER long-context evaluation suite. The results are presented in Table 4.

Our experiments reveal that, compared to the base model Qwen3.5-32B-Instruct, models trained on a predominantly math problem dataset—both the official Qwen checkpoint QwQ-32B-Preview and models fine-tuned on open-

source datasets—exhibit decreased performance. This decline becomes more pronounced as the input length increases from 32K to 64K tokens. However, models fine-tuned with URT are exposed to numerous long prompt inputs during training. Consequently, compared to direct SFT, URT-SFT does not suffer significant performance degradation on long-context tasks and improves performance on long-context tasks by over 10 percentage points compared to QwQ-32B-Preview.

In conclusion, By decomposing long-chain reasoning into manageable segments embedded within prompts, URT-SFT not only simplifies the learning process but also capitalizes on the models’ ability to handle extended contexts, leading to improved accuracy and performance.

4.4 Analysis

Performance Gains Across Different Difficulty Levels To investigate the scenarios in which URT-SFT enhances standard SFT, we constructed a diverse test set by randomly sampling 200 instances from the MATH dataset across difficulty levels 1 to 5 and 30 instances from AIME 2024, resulting in a test set with varying levels of difficulty. Specifically, **Level 1** includes problems from MATH difficulty levels 1, 2, and 3; **Level 2** comprises MATH difficulty level 4 problems; **Level 3** consists of

Table 4: Long context results of long reasoning models. We test the models on 3 categories: Needle-In-A-Haystack (NIAH), Variable Tracing (VT) and Question Answering (QA) from RULER (Hsieh et al., 2024). We conduct tests using input lengths of 32K and 64K tokens. The demonstrations of the three tasks defined in RULER are shown in Appendix Table 6 and 7.

Models	RULER			Avg.
	NIAH	VT	QA	
Context length = 32K				
Qwen2.5-32B-Instruct	98.0	86.4	62.0	82.1
QwQ-32B-Preview	64.0	82.8	60.0	68.9
Qwen2.5 + SFT	92.0	89.0	50.0	77.0
Qwen2.5 + URT-SFT	95.0	90.4	59.0	81.5
Context length = 64K				
Qwen2.5-32B-Instruct	71.0	67.8	56.0	64.9
QwQ-32B-Preview	52.0	57.4	37.0	48.8
Qwen2.5 + SFT	55.0	70.9	31.0	52.3
Qwen2.5 + URT-SFT	62.0	74.8	45.0	60.6

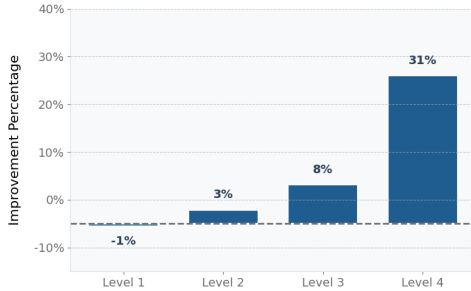


Figure 4: URT-SFT shows a greater performance improvement over SFT on more difficult math problems.

MATH difficulty level 5 problems; and **Level 4** contains the 30 AIME 2024 problems. The performance improvements of URT-over SFT across these different difficulty levels are illustrated in Figure 4. Our method demonstrates superior performance on more challenging tasks, highlighting its effectiveness in handling problems of increased complexity.

Reinforcement Learning To evaluate the orthogonality between URT and Reinforcement Learning (RL) fine-tuning, we conduct complementary experiments using Group Relative Policy Optimization (GRPO) (DeepSeek-AI et al., 2025). Specifically, we fine-tune our models for 1 epoch on the STILL2 training set with a rollout size of 8 and a maximum sequence length of 16,384 tokens per trajectory, following the default configurations of Verl (Sheng et al., 2024) with 8 NVIDIA A100

GPUs. Our results (Figure 5) indicate that RL scaling and URT are orthogonal, with RL further enhancing URT’s performance. Notably, URT demonstrates significantly higher training efficiency compared to RL scaling. For instance, training a 32B model with GRPO consumes over **12 days** on 8 A100 GPUs, suggesting that knowledge distillation may be a more practical alternative for smaller models with academic-level resources.

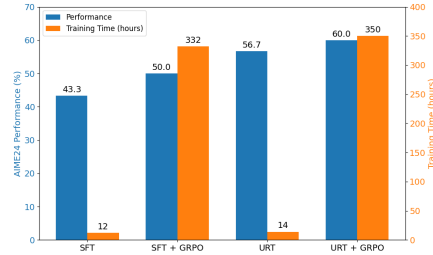


Figure 5: Training time and performance on AIME24 for SFT and URT models further trained with the GRPO.

Inference Time Scaling In our SFT-trained models, we observe inference time scaling behavior similar to that of the o1 model. Specifically, as the number of tokens processed during inference increases, the performance continues to improve. In all experiments, the models were run with a maximum new tokens of 16K.

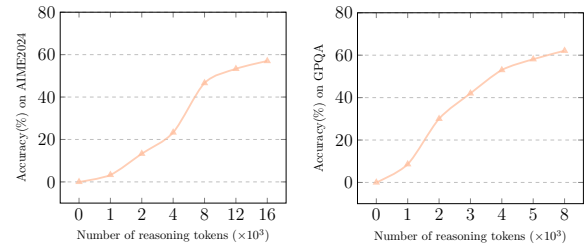


Figure 6: Test-time scaling with URT models on GPQA and AIME24.

5 Conclusion

This paper introduces a new fine-tuning framework called Understanding-to-Reasoning Transition (URT) fine-tuning, which aims to improve the ability of large language models to generate long and coherent CoT reasoning. The core idea is to gradually transition the model’s focus from understanding CoTs to generating them. This is achieved by incorporating partial reasoning steps into the input context during fine-tuning. The experimental results demonstrate that URT fine-tuning outperforms standard fine-tuning methods, especially on challenging reasoning benchmarks.

Limitations

One limitation of this work is that we have only explored the effectiveness of a training method that gradually transfers from understanding to reasoning within the fine-tuning phase. Due to computational resource constraints, we have not validated the efficacy of our approach within the reinforcement learning framework. Additionally, in terms of long-context understanding, when the input length exceeds 64K tokens, the performance of models trained using URT declines compared to the base model. To further enhance the long reasoning model’s ability to handle lengthy prompts, it may be necessary to incorporate more high-quality long prompt data, rather than relying solely on a training set predominantly consisting of mathematical problems.

References

Reza Yazdani Aminabadi, Samyam Rajbhandari, Ammar Ahmad Awan, Cheng Li, Du Li, Elton Zheng, Olatunji Ruwase, Shaden Smith, Minjia Zhang, Jeff Rasley, et al. 2022. Deepspeed-inference: enabling efficient inference of transformer models at unprecedented scale. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15. IEEE.

Chenxin An, Shansan Gong, Ming Zhong, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2023. L-eval: Instituting standardized evaluation for long context language models. *arXiv preprint arXiv:2307.11088*.

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, and Jian-Guang Lou. 2024. *Make your llm fully utilize the context*. *Preprint*, arXiv:2404.16811.

Anthropic. 2023. *Introducing 100K Context Windows*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. *Qwen technical report*. *Preprint*, arXiv:2309.16609.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2025. *Longbench v2: Towards deeper understanding*

and reasoning on realistic long-context multitasks. *Preprint*, arXiv:2412.15204.

Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. *Longwriter: Unleashing 10,000+ word generation from long context llms*. *Preprint*, arXiv:2408.07055.

Mingda Chen, Zewei Chu, Sam Wiseman, and Kevin Gimpel. 2022. *SummScreen: A dataset for abstractive screenplay summarization*. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8602–8615, Dublin, Ireland. Association for Computational Linguistics.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023. Longlora: Efficient fine-tuning of long-context large language models. *arXiv:2309.12307*.

Tri Dao. 2023. *Flashattention-2: Faster attention with better parallelism and work partitioning*. *Preprint*, arXiv:2307.08691.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. *A dataset of information-seeking questions and answers anchored in research papers*. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, Online. Association for Computational Linguistics.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shutong Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanbiao Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin

691	Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li,	Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul	747
692	Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin,	Arora, Steven Basart, Eric Tang, Dawn Song, and	748
693	Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxi-	Jacob Steinhardt. 2021b. Measuring mathematical	749
694	ang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang,	problem solving with the math dataset . <i>Preprint</i> ,	750
695	Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang	arXiv:2103.03874.	751
696	Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng		
697	Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi,	Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shan-	752
698	Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang,	tanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang,	753
699	Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo,	and Boris Ginsburg. 2024. Ruler: What’s the real	754
700	Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yu-	context size of your long-context language models?	755
701	jia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You,	<i>Preprint</i> , arXiv:2404.06654.	756
702	Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu,		
703	Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu,	Zhongzhen Huang, Gui Geng, Shengyi Hua, Zhen	757
704	Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan,	Huang, Haoyang Zou, Shaoting Zhang, Pengfei Liu,	758
705	Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean	and Xiaofan Zhang. 2025. O1 replication journey –	759
706	Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao,	part 3: Inference-time scaling for medical reasoning .	760
707	Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zi-	<i>Preprint</i> , arXiv:2501.06458.	761
708	jia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song,		
709	Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu	Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia	762
710	Zhang, and Zhen Zhang. 2025. Deepseek-r1: Incen-	Yan, Tianjun Zhang, Sida Wang, Armando Solar-	763
711	tivizing reasoning capability in llms via reinforce-	Lezama, Koushik Sen, and Ion Stoica. 2024. Live-	764
712	ment learning . <i>Preprint</i> , arXiv:2501.12948.	codebench: Holistic and contamination free evalu-	765
713	DeepSeek Team. 2024. Deepseek r1 .	ation of large language models for code . <i>Preprint</i> ,	766
		arXiv:2403.07974.	767
714	Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Han-	Ziyan Jiang, Xueguang Ma, and Wenhui Chen. 2024.	768
715	naneh Hajishirzi, Yoon Kim, and Hao Peng. 2024.	Longrag: Enhancing retrieval-augmented gener-	769
716	Data engineering for scaling language models to 128k	ation with long-context llms. <i>arXiv preprint</i>	770
717	context . <i>Preprint</i> , arXiv:2402.10171.	arXiv:2406.15319.	771
718	Chaochen Gao, Xing Wu, Qingfang Fu, and Songlin	Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng,	772
719	Hu. 2024a. Quest: Query-centric data synthesis ap-	Hanyi Fang, and Peter Szolovits. 2020. What dis-	773
720	proach for long-context scaling of large language	ease does this patient have? a large-scale open do-	774
721	model . <i>ArXiv</i> , abs/2405.19846.	main question answering dataset from medical exams .	775
722		<i>Preprint</i> , arXiv:2009.13081.	776
723	Tianyu Gao, Alexander Wettig, Howard Yen, and Danqi		
724	Chen. 2024b. How to train long-context language	Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris	777
	models (effectively) . <i>Preprint</i> , arXiv:2410.02660.	Dyer, Karl Moritz Hermann, Gábor Melis, and Ed-	778
725	gkamradt. 2023. Llmtest_needleinahaystack:	ward Grefenstette. 2018. The NarrativeQA reading	779
726	Doing simple retrieval from llm models .	comprehension challenge . <i>Transactions of the Asso-</i>	780
727	https://github.com/gkamradt/LLMTest_	<i>ciation for Computational Linguistics</i> , 6:317–328.	781
728	NeedleInAHaystack/tree/main . [Online; ac-		
729	cessed 29-December-2023].	Nithin Rao Koluguri, Travis Bartley, Hainan Xu, Olek-	782
730	Google Team. 2024. Gemini 2.0 flash thinking mode	sii Hrinchuk, Jagadeesh Balam, Boris Ginsburg, and	783
731	(gemini-2.0-flash-thinking-exp-1219) .	Georg Kucsko. 2024. Longer is (not necessarily)	784
732	Aryan Gulati, Brando Miranda, Eric Chen, Emily Xia,	stronger: Punctuated long-sequence training for en-	785
733	Kai Fronsdal, Bruno de Moraes Dumont, and Sanmi	hanced speech recognition and translation. In <i>2024</i>	786
734	Koyejo. 2024. Putnam-AXIOM: A functional and	<i>IEEE Spoken Language Technology Workshop (SLT)</i> ,	787
735	static benchmark for measuring higher level mathe-	pages 255–262. IEEE.	788
736	matical reasoning . In <i>The 4th Workshop on Mathe-</i>		
737	matical Reasoning and AI at NeurIPS’24 .	Philippe Laban, Alexander R Fabbri, Caiming Xiong,	789
738	Wei Han, Pan Zhou, Soujanya Poria, and Shuicheng	and Chien-Sheng Wu. 2024. Summary of a haystack:	790
739	Yan. 2024. Two are better than one: Context window	A challenge to long-context llms and rag systems.	791
740	extension with multi-grained self-injection. <i>arXiv</i>	<i>arXiv preprint arXiv:2407.01370</i> .	792
741	<i>preprint arXiv:2410.19318</i> .		
742	Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul	Ehsan Latif, Yifan Zhou, Shuchen Guo, Yizhu Gao,	793
743	Arora, Steven Basart, Eric Tang, Dawn Song, and	Lehong Shi, Matthew Nayaaba, Gyeonggeon Lee,	794
744	Jacob Steinhardt. 2021a. Measuring mathemati-	Liang Zhang, Arne Bewersdorff, Luyang Fang, Xi-	795
745	cal problem solving with the math dataset . <i>arXiv</i>	antong Yang, Huaqin Zhao, Hanqi Jiang, Haoran Lu,	796
746	<i>preprint arXiv:2103.03874</i> .	Jiaxi Li, Jichao Yu, Weihang You, Zhengliang Liu,	797
		Vincent Shung Liu, Hui Wang, Zihao Wu, Jin Lu,	798
		Fei Dou, Ping Ma, Ninghao Liu, Tianming Liu, and	799
		Xiaoming Zhai. 2024. A systematic assessment of	800
		openai o1-preview for higher order thinking in edu-	801
		cation . <i>Preprint</i> , arXiv:2410.21287.	802

- Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. [Numinamath](#).
- Siheng Li, Cheng Yang, Zesen Cheng, Lemao Liu, Mo Yu, Yujiu Yang, and Wai Lam. 2024. Large language models can self-improve in long-context reasoning. *arXiv preprint arXiv:2411.08147*.
- Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. [Let’s verify step by step](#). *Preprint*, arXiv:2305.20050.
- Hongzhan Lin, Ang Lv, Yang Song, Hengshu Zhu, Rui Yan, et al. 2025. Mixture of in-context experts enhance llms’ long context awareness. *Advances in Neural Information Processing Systems*, 37:79573–79596.
- Gabrielle Kaili-May Liu, Bowen Shi, Avi Caciularu, Idan Szpektor, and Arman Cohan. 2024. Mdcure: A scalable pipeline for multi-document instruction-following. *arXiv preprint arXiv:2410.23463*.
- Llama Team. 2024. [The llama 3 herd of models](#). *CoRR*, abs/2407.21783.
- Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). *Preprint*, arXiv:1711.05101.
- MAA. 2024. [American invitational mathematics examination - aime](#). In *American Invitational Mathematics Examination - AIME 2024*.
- Nikolay Mikhaylovskiy. 2023. [Long story generation challenge](#). In *Proceedings of the 16th International Natural Language Generation Conference: Generation Challenges*, pages 10–16, Prague, Czechia. Association for Computational Linguistics.
- 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 Ji-Rong Wen. 2024. [Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems](#). *Preprint*, arXiv:2412.09413.
- Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. [s1: Simple test-time scaling](#). *Preprint*, arXiv:2501.19393.
- NovaSky Team. 2025. [Sky-t1: Fully open-source reasoning model with o1-preview performance in \\$450 budget](#). Accessed: 2025-01-09.
- Skywork o1 Team. 2024. [Skywork-o1 open series](#). <https://huggingface.co/Skywork>.
- OpenAI. 2023. [Gpt-4 technical report](#). *Preprint*, arXiv:2303.08774.
- OpenAI. 2024. [Learning to reason with llms](#).
- Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. 2025. [Tinyzero](https://github.com/Jiayi-Pan/TinyZero). <https://github.com/Jiayi-Pan/TinyZero>. Accessed: 2025-01-24.
- Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. [Yarn: Efficient context window extension of large language models](#). *Preprint*, arXiv:2309.00071.
- Qwen, :, 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, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. [Qwen2.5 technical report](#). *Preprint*, arXiv:2412.15115.
- Qwen Team. 2024. [Qwq: Reflect deeply on the boundaries of the unknown](#).
- Machel Reid et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv:2403.05530*.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. [Gpqa: A graduate-level google-proof q&a benchmark](#). *Preprint*, arXiv:2311.12022.
- Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, and Omer Levy. 2022. [SCROLLS: Standardized CompaRison over long language sequences](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 12007–12021, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*.
- Mingyang Song, Mao Zheng, and Xuan Luo. 2025. Counting-stars: A multi-evidence, position-aware, and scalable benchmark for evaluating long-context large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3753–3763.

909	Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them . <i>Preprint</i> , arXiv:2210.09261.	964
910		965
911		966
912		967
913		968
914		969
915	Open Thoughts Team. 2025. Open Thoughts.	
916	Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models . <i>Preprint</i> , arXiv:2302.13971.	
917		
918		
919		
920		
921		
922		
923	Jiaan Wang, Fandong Meng, Yunlong Liang, and Jie Zhou. 2024. Drt-ol: Optimized deep reasoning translation via long chain-of-thought . <i>Preprint</i> , arXiv:2412.17498.	
924		
925		
926		
927	Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023. How far can camels go? exploring the state of instruction tuning on open resources . <i>Preprint</i> , arXiv:2306.04751.	
928		
929		
930		
931		
932		
933	Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models . <i>Preprint</i> , arXiv:2201.11903.	
934		
935		
936		
937		
938	Wenhao Wu, Yizhong Wang, Yao Fu, Xiang Yue, Dawei Zhu, and Sujian Li. 2024. Long context alignment with short instructions and synthesized positions . <i>Preprint</i> , arXiv:2405.03939.	
939		
940		
941		
942	Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, Madian Khabisa, Han Fang, Yashar Mehdad, Sharan Narang, Kshitiz Malik, Angela Fan, Shruti Bhosale, Sergey Edunov, Mike Lewis, Sinong Wang, and Hao Ma. 2023. Effective long-context scaling of foundation models . <i>CoRR</i> , abs/2309.16039.	
943		
944		
945		
946		
947		
948		
949		
950	Haotian Xu, Xing Wu, Weinong Wang, Zhongzhi Li, Da Zheng, Boyuan Chen, Yi Hu, Shijia Kang, Jiaming Ji, Yingying Zhang, Zhijiang Guo, Yaodong Yang, Muhan Zhang, and Debing Zhang. 2025. Redstar: Does scaling long-cot data unlock better slow-reasoning systems? <i>Preprint</i> , arXiv:2501.11284.	
951		
952		
953		
954		
955		
956	Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models . <i>Preprint</i> , arXiv:2305.10601.	
957		
958		
959		
960		
961	Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning . <i>Preprint</i> , arXiv:2502.03387.	
962		
963		
	Di Zhang, Jianbo Wu, Jingdi Lei, Tong Che, Jiatong Li, Tong Xie, Xiaoshui Huang, Shufei Zhang, Marco Pavone, Yuqiang Li, Wanli Ouyang, and Dongzhan Zhou. 2024a. Llama-berry: Pairwise optimization for ol-like olympiad-level mathematical reasoning . <i>Preprint</i> , arXiv:2410.02884.	970
		971
	Hugh Zhang and Celia Chen. 2024. Test-time compute scaling laws .	
	Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2024b. ∞bench: Extending long context evaluation beyond 100k tokens . <i>Preprint</i> , arXiv:2402.13718.	972
		973
		974
		975
		976
		977
	Yuxiang Zhang, Shangxi Wu, Yuqi Yang, Jiangming Shu, Jinlin Xiao, Chao Kong, and Jitao Sang. 2024c. ol-coder: an ol replication for coding . <i>Preprint</i> , arXiv:2412.00154.	978
		979
		980
		981
	Liang Zhao, Tianwen Wei, Liang Zeng, Cheng Cheng, Liu Yang, Peng Cheng, Lijie Wang, Chenxia Li, Xuejie Wu, Bo Zhu, Yimeng Gan, Rui Hu, Shuicheng Yan, Han Fang, and Yahui Zhou. 2024a. Longskywork: A training recipe for efficiently extending context length in large language models . <i>Preprint</i> , arXiv:2406.00605.	982
		983
		984
		985
		986
		987
		988
	Yu Zhao, Huifeng Yin, Bo Zeng, Hao Wang, Tianqi Shi, Chenyang Lyu, Longyue Wang, Weihua Luo, and Kaifu Zhang. 2024b. Marco-ol: Towards open reasoning models for open-ended solutions . <i>Preprint</i> , arXiv:2411.14405.	989
		990
		991
		992
		993
	Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyang Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models . <i>arXiv preprint arXiv:2403.13372</i> .	994
		995
		996
		997
		998
	Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021. QMSum: A new benchmark for query-based multi-domain meeting summarization . In <i>Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies</i> , pages 5905–5921, Online. Association for Computational Linguistics.	999
		1000
		1001
		1002
		1003
		1004
		1005
		1006
		1007
		1008
	Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024. Language agent tree search unifies reasoning acting and planning in language models . <i>Preprint</i> , arXiv:2310.04406.	1009
		1010
		1011
		1012
		1013
	Dawei Zhu, Nan Yang, Liang Wang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. 2023. Pose: Efficient context window extension of llms via positional skip-wise training . <i>Preprint</i> , arXiv:2309.10400.	1014
		1015
		1016
		1017

A Appendix

A.1 Benchmarks and Baseline Models

Benchmarks We utilize three widely-recognized reasoning benchmarks to assess the problem-solving skills of our models: (1) AIME24 comprises 30 mathematical problems from the 2024 American Invitational Mathematics Examination. (2) MATH500 (Hendrycks et al., 2021b; Lightman et al., 2023) is a curated subset of 500 competition-level problems. (3) GPQA Diamond (Rein et al., 2023) comprises 198 PhD-level STEM questions across Biology, Chemistry, and Physics. We use this benchmark to assess our model’s reasoning ability across STEM domains. We also include MedQA (Jin et al., 2020), LiveCodeBench-Medium (Jain et al., 2024) (24.08-24.1), BBH (Suzgun et al., 2022), and Putnam-AXIOM (Gulati et al., 2024) to evaluate our method across broader domains, covering medical exams, coding, and commonsense. Since most datasets for long CoT training are dominated by short inputs, we incorporate the RULER (Hsieh et al., 2024) evaluation suite to assess whether models’ long-context understanding abilities are affected. This suite features long-document comprehension tasks with customizable input lengths.

Baselines We compare our models against several strong baselines to contextualize performance: **o1** and **o1-preview** (OpenAI, 2024), commercial models by OpenAI known for generating long reasoning traces; **r1** and **r1-preview** (DeepSeek Team, 2024), developed by DeepSeek with performance comparable to o1; **QwQ-32B-preview** (Qwen Team, 2024), an open-weight reasoning model from the Qwen Team; **Sky-T1-32B-Preview** (NovaSky Team, 2025), trained on reasoning data distilled from QwQ-32B-preview; **s1-32B-BF**, trained on data distilled from Google Gemini 2.0 Flash Thinking (Google Team, 2024) with budget forcing to encourage longer reasoning paths; **STILL2-32B** (Min et al., 2024), utilizing the same training dataset as our model with additional refinements and reinforcement learning; and **Open-o1-Llama3.1-8B**, an o1-like model based on Llama3.1-8B-Instruct.

Table 5: The example input prompt and model output for the reasoning benchmarks used in this paper.

MATH500	
<i>Input prompt</i>	<p>Let a be a positive real number such that all the roots of</p> $x^3 + ax^2 + ax + 1 = 0$ <p>are real. Find the smallest possible value of a</p>
<i>Model reponse</i>	<p>Let's tackle this problem step by step. We have the cubic equation:</p> $x^3 + ax^2 + ax + 1 = 0$ <p>And we're told that all its roots are real, with a being a positive real number. We need to find the smallest possible value of a. First, I know that for a cubic equation of the form $x^3 + bx^2 + cx + d = 0$</p> <p>...</p> <p>Thus, the smallest possible value of a is: <code>boxed{3}</code></p>
AIME 2024	
<i>Input prompt</i>	<p>Find the largest possible real part of</p> $(75 + 117i)z + \frac{96 + 144i}{z}$ <p>where z is a complex number with $z = 4$</p>
<i>Model reponse</i>	<p>Alright, so I've got this problem here. It says: Find the largest possible real part of</p> $(75 + 117i)z + \frac{96 + 144i}{z},$ <p>where z is a complex number with $z = 4$. Okay, let's break this down. First off, z is a complex number with magnitude 4. So, z can be written in polar form as $z = 4e^{i\theta}$, where θ is the argument of z. But maybe it's easier to work with z in terms of its real and imaginary parts. Let's say $z = a + bi$, where a and b are real numbers, and $a^2 + b^2 = 16$, since $z = 4$</p> <p>...</p> <p>Therefore, the maximum value of the real part is: <code>boxed{540}</code>.</p>
GPQA Diamond	
<i>Input prompt</i>	<p>What is the energy of the Relativistic Heavy Ion Collider (RHIC) so that the speed of the nucleus X is equal to $0.96c$? Knowing that X is defined as Li with $A=6$.</p> <p>A. 20 GeV B. 10 GeV C. 36 GeV D. 49 GeV</p>
<i>Model reponse</i>	<p>Iright, I've got this problem about the Relativistic Heavy Ion Collider, or RHIC, and I need to find out what energy it should have so that the speed of a lithium nucleus, specifically Li with $A=6$, is equal to 0.96 times the speed of light. The options are 20 GeV, 10 GeV, 36 GeV, and 49 GeV. First off, I know that in relativity, when objects move at speeds close to the speed of light, their energy isn't just mc^2; there's additional kinetic energy involved. The total energy of a relativistic particle is given by $E = \gamma mc^2$, where γ is the Lorentz factor. The Lorentz factor, γ, is $\frac{1}{\sqrt{1 - \frac{v^2}{c^2}}}$.</p> <p>So, if the speed v is $0.96c$, then v/c is 0.96. Let me calculate γ.</p> $\gamma = \frac{1}{\sqrt{1 - (0.96)^2}}$ <p>...</p> <p>This value is closest to 20 GeV, so the correct answer is: <code>boxed{A}</code></p>

Table 6: The example input prompt and model output for the Needle-in-a-Haystack and Variable Tracing in RULER.

Needle-in-a-Haystack (multi-needle)	
<i>Input prompt</i>	<p>July 2006I've discovered a handy test for figuring out what you're addicted to. Imagine you were going to spend the weekend at a friend's house on a little island off the coast of Maine. There are no shops on the island and you won't be able to leave while you're there. Also, you've never been to this house before, so you can't assume it will have more than any house might. What, besides clothes and toiletries, do you make a point of packing? That's what you're addicted to... The UUID for bd9645ff-0667-1ad1-1e80-b17fa7f1599d is 9dde507b-0676-9ad1-1e80-317fa3b1799c you can't expect an idea to be novel to everyone. Any insight that you have will probably have already been had by at least one of the world's 7 billion people. But it's sufficient if an idea is novel to a lot of readers. Ditto for correctness, importance, and strength. In effect the four components But with Lisp our development cycle was so fast that we could sometimes duplicate a new feature within a day or two of a competitor announcing it in a press release. By the time journalists covering the press release got round to The UUID for b5d648fb-0666-1ad3-5c80-39ff23b379ad is: bcd640fb-0667-1ad5-1c88-337fa3b0699d nThere is a founder community just as there's a VC community. They all know one another, and techniques spread rapidly between them. If one tries a new programming language or a new hosting provider and gets good results, 6 months later half of...</p> <p>Please find the UUID for bd9645ff-0667-1ad1-1e80-b17fa7f1599d from the text above</p>
<i>Model reponse</i>	<p>Alright, I need to find the UUID for bd9645ff-0667-1ad1-1e80-b17fa7f1599d from the provided document. Let me look through the text carefully. I see that there are several UUIDs mentioned, each associated with a specific UUID format. I need to find the one that matches bd9645ff-0667-1ad1-1e80-b17fa7f1599d and note its corresponding UUID. Looking at the first one: The UUID for bd9645ff-0667-1ad1-1e80-b17fa7f1599d is 9dde507b-0676-9ad1-1e80-317fa3b1799c. So, it seems like the UUID for bd9645ff-0667-1ad1-1e80-b17fa7f1599d is 9dde507b-0676-9ad1-1e80-317fa3b1799c. I should double-check to make sure I don't miss any other occurrences. Looking further down, I see another UUID mentioned: The UUID for b5d648fb-0666-1ad3-5c80-39ff23b379ad is: bcd640fb-0667-1ad5-1c88-337fa3b0699d. And another one: The UUID for fdc644fb-1667-1ad1-0d80-337fe3b7799d is: bdf6c05b-0427-dad1-0c00-317fa321699d...</p> <p>Thus, the final answer is: boxed{9dde507b-0676-9ad1-1e80-317fa3b1799c}</p>
Variable Tracing	
<i>Input prompt</i>	<p>July 2006I've discovered a handy test for figuring out what you're addicted to. Imagine you were going to spend the weekend at a friend's house on a little island off the coast of Maine. There are no shops on the island and you won't be able to leave while you're there. Also, you've never been to this house before, so you can't assume it will have more than any house might... var GNEXW = 62265 they're going to issue a public report tracking how this specific tranche of money is spent, NFTs are a new territory... var HQPDT = GNEXW you can't expect an idea to be novel to everyone. Any insight that you have will probably have already been had by at least one of the world's 7 billion people. But it's sufficient if an idea is novel to a lot of readers. Ditto for correctness, importance, and strength. In effect the four components var OUNAI = HQPDT But with Lisp our development cycle was so fast that we could sometimes duplicate a new feature within a day or two of a competitor announcing it in a press release. By the time journalists covering the press release got round to var AYWVH = OUNAI nThere is a founder community just as there's a VC community. They all know one another, and techniques spread rapidly between them. var BWYCM = AYWVH If one tries a new programming language or a new hosting provider and gets good results, 6 months later half of...</p> <p>Please find all the 5 variables that are assigned the value 62265 in the text above.</p>
<i>Model reponse</i>	<p>Alright, I need to find all the variables that are assigned the value 62265 in this text. I'll look for lines that have var followed by a variable name, an equals sign, and then 62265. First, I see var GNEXW = 62265. So GNEXW is one variable assigned to 62265. Next, I look further down and find var HQPDT = GNEXW. Wait, HQPDT is being assigned the value of GNEXW, which is 62265. So HQPDT is another variable with the value 62265. Then, I see var BTTDM = 99634. That's not 62265, so I skip that. Later, there's var OUNAI = HQPDT. Since HQPDT is 62265, OUNAI is also 62265... I need to make sure I didn't miss any or count any that shouldn't be included. Let me double-check.</p> <ul style="list-style-type: none"> - GNEXW = 62265 (direct assignment) - HQPDT = GNEXW (GNEXW is 62265) - OUNAI = HQPDT (HQPDT is 62265) - AYWVH = OUNAI (OUNAI is 62265) - BWYCM = AYWVH (AYWVH is 62265) <p>Looks correct. These are the five variables assigned to 62265. Thus, the final answer is: boxed{GNEXW, HQPDT, OUNAI, AYWVH, BWYCM }</p>

Table 7: An example of the input prompt and corresponding model output for the long-context QA task in RULER.

Long context QA	
<i>Input prompt</i>	<p>The Normans had a profound effect on Irish culture and history after their invasion at Bannow Bay in 1169. Initially the Normans maintained a distinct culture and ethnicity. Yet, with time, they came to be subsumed into Irish culture to the point that it has been said that they became "more Irish than the Irish themselves." The Normans settled mostly in an area in the east of Ireland, later known as the Pale, and also built many fine castles and settlements, including Trim Castle and Dublin Castle. Both cultures intermixed, borrowing from each other's language, culture and outlook. Norman descendants today can be recognised by their surnames. Names such as French, (De) Roche, Devereux, D'Arcy, Treacy and Lacy are particularly common in the southeast of Ireland, especially in the southern part of County Wexford where the first Norman settlements were established. Other Norman names such as Furlong predominate there. Another common Norman-Irish name was Morell (Murrell) derived from the French Norman name Morel. Other names beginning with Fitz (from the Norman for son) indicate Norman ancestry. These included Fitzgerald, FitzGibbons (Gibbons) dynasty, Fitzmaurice. Other families bearing such surnames as Barry (de Barra) and De Búrca (Burke) are also of Norman extraction. rior to European settlement, the area now constituting Victoria was inhabited by a large number of ..(lengthy documents containing the background for the question)... Victoria was officially created a separate colony in 1851, and achieved self-government in 1855. The Victorian gold rush in the 1850s and 1860s significantly increased both the population and wealth of the colony, and by the Federation of Australia in 1901, Melbourne had become the largest city and leading financial centre in Australasia. Melbourne also served as capital of Australia until the construction of Canberra in 1927, with the Federal Parliament meeting in Melbourne's Parliament House and all principal offices of the federal government being based in Melbourne. Before the St. Elizabeth's flood (1421), the Meuse flowed just south of today's line Merwede-Oude Maas to the North Sea and formed an archipelago-like estuary with Waal and Lek. This system of numerous bays, estuary-like extended rivers, many islands and constant changes of the coastline, is hard to imagine today. From 1421 to 1904, the Meuse and Waal merged further upstream at Gorinchem to form Merwede. For flood protection reasons, the Meuse was separated from the Waal through a lock and diverted into a new outlet called "Bergse Maas", then Amer and then flows into the former bay Hollands Diep.</p> <p>What river originally bounded the Duchy?</p>
<i>Model reponse</i>	<p>Alright, I'm trying to figure out which river originally bounded the Duchy. I remember that the Duchy of Normandy was established in the 10th century, and it was located in what is now northern France. I think the river in question is the Seine because I recall that Normandy extended down to the Seine River. But I'm not entirely sure, maybe it's another river? Let me think. I know that the Duchy of Normandy was founded by Rollo, a Viking leader, and it was granted to him by the French king in the Treaty of Saint-Clair-sur-Epte in 911. The treaty specified that Rollo and his men could settle in the area between the river Epte and the Atlantic coast. So, the Epte River might have been a boundary, but I don't think that's the main river that bounded the Duchy... Therefore, the river that originally bounded the Duchy of Normandy was the boxed{Epte River}</p>

Table 8: Error bars for the main results. To assess the statistical significance of our results in Table 2, we conduct multiple runs of the model using top-p decoding 16 times with a temperature of 0.6 and top-p=0.95 using our 32B model.

Dataset	SFT		URT	
	Mean	Std Dev	Mean	Std Dev
AIME	45.6	4.85	57.1	4.93
GPQA	55.6	1.72	62.0	1.76
MATH500	87.4	1.04	89.8	0.98