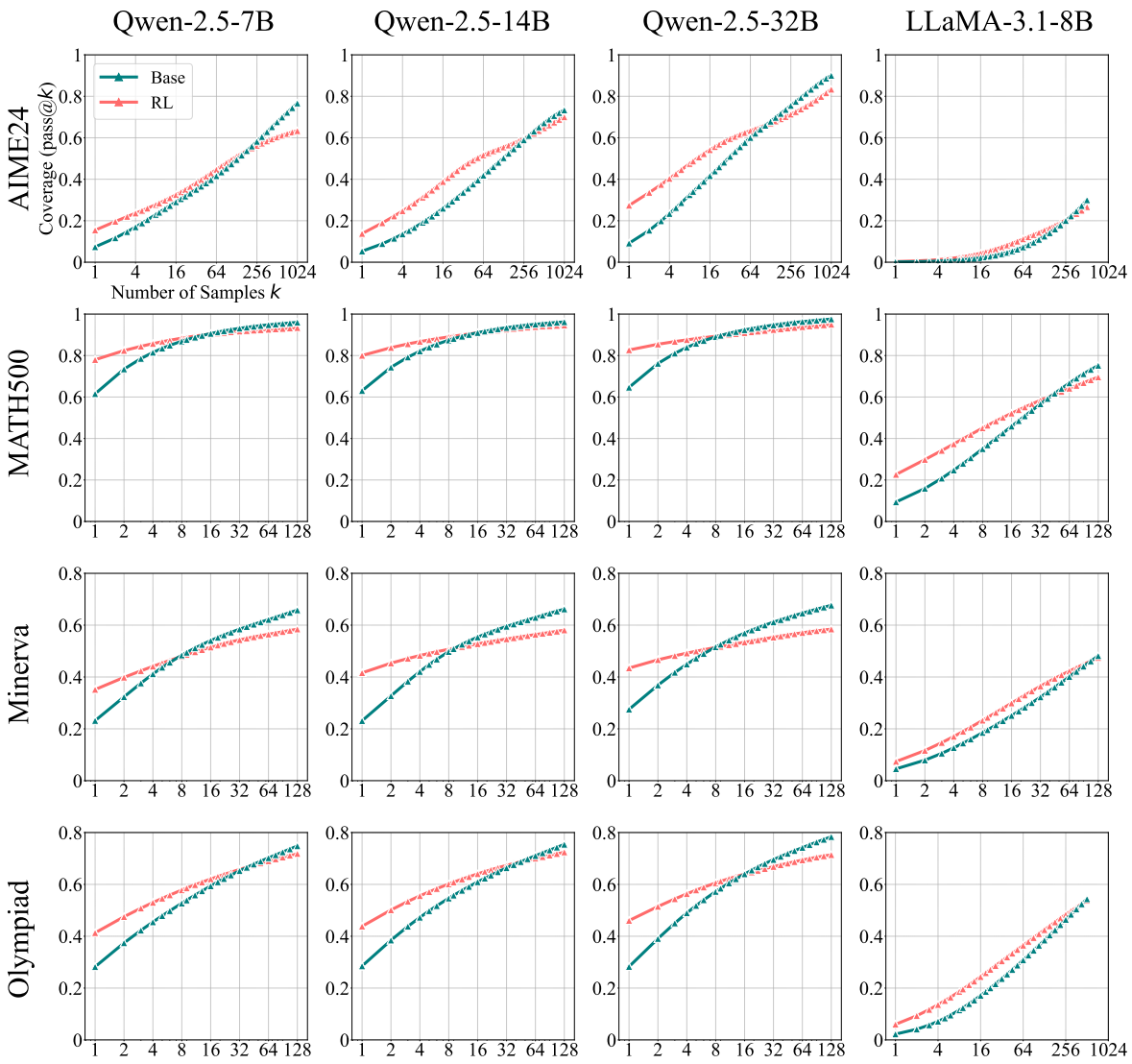

Figure 2: Pass@k curves of base models and their RLVR-trained counterparts across multiple mathematical benchmarks. When k is small, RL-trained models outperform their base versions. However, as k increases to the tens or hundreds, base models consistently catch up and surpass RL-trained models. More results on GSM8K and AMC23 can be found at Figure 10.

KL divergence term is optionally applied, to constrain the model from deviating too far from the original policy. More algorithms are introduced in Section C.5.

Policy Gradient. PPO and its variants belong to the policy gradient class of RL (Williams, 1992; Sutton et al., 1998). These methods learn exclusively from *on-policy samples*, *i.e.*, samples generated by the current LLM. In the context of verifiable rewards, the training objective generally *maximizes the log-likelihood of samples with correct answers and minimizes the likelihood of those with incorrect answers.*

Zero RL Training applies RL directly to the base model without any supervised fine-tuning (SFT) (Guo et al., 2025). To clearly study the effect of RLVR, we follow this zero-RL setting for all math tasks using pretrained models as start model. However, for coding and visual reasoning tasks, open-source work typically uses instruction-tuned models as starting points, primarily due to the training instability and limited effectiveness of using a pure zero-RL setting. Following this convention, we compare the finetuned model with its RLVR-trained counterpart to focus solely on the effect of RLVR.

2.2. Metrics for LLM Reasoning Capacity Boundary

Pass@k Metrics. Accurately measuring the reasoning ability boundary of base and RL models is challenging, as methods like greedy decoding or the average of nucleus samplings (Holtzman et al., 2020) only reflect average-case performance. To accurately measure the reasoning ability boundary, we extend the commonly used pass@k metric from code generation (Chen et al., 2021) to all tasks with verifiable rewards. Given a problem, we sample k outputs from the model. The pass@k value for this question is 1 if at least one of the k samples passes verification; otherwise, it is 0. The average pass@k value over the dataset reflects the proportion of problems in the dataset that the model can solve within k trials, providing a rigorous evaluation of the reasoning capacity coverage of LLMs. We adopt an unbiased, low-variance estimator for computing to calculate pass@k, as detailed in Section A.2.

Comparison with Best-of-N and Majority Voting. Best-of-N (Cobbe et al., 2021) and majority voting are practical methods for selecting correct answers, but they may overlook a model's full reasoning potential. In contrast, we use pass@k *not to assess practical utility* but to investigate the boundaries of reasoning capacity. If a model produces a correct solution in any of the k samples, we treat the problem as within its potential scope. Thus, if RL enhances reasoning, the RL-trained model should succeed in more such problems than the base model. Methods like Best-of-N or majority voting may miss these successes if the correct answer is not selected by the verifier or voting.

Random Guessing Issue. For *coding* tasks, where a compiler and predefined unit test cases are used as verifiers, the pass@k value can accurately reflect whether the model can solve the problem. In *mathematics*, the issue of "guessing" can become pronounced as k increases, where a model may generate an incorrect CoT but still accidentally arrive at the correct answer. To address this, we manually check the correctness of CoT for a subset of model outputs as detailed in Section 3.1. By combining results on math with manually checking and coding, we rigorously evaluate the scope of LLM's reasoning capacity. Another caveat is that, with an astronomically large k, even uniform sampling over the token dictionary would stumble upon the correct reasoning path–though this is infeasible within today's time and compute budgets. Crucially, we find that the base model already produces correct outputs at realistic values ($k = 128$ or 1024), well within practical resource limits.

Table 1: Experimental setup for assessing RLVR's effect on the reasoning boundaries of LLMs.

Task	Start Model	RL Framework	RL Algorithm(s)	Benchmark(s)
Mathematics	LLaMA-3.1-8B Qwen2.5-7B/14B/32B-Base Qwen2.5-Math-7B	SimpleRLZoo Oat-Zero DAPO	GRPO	GSM8K, MATH500 Minerva, Olympiad AIME24, AMC23
Code Generation	Qwen2.5-7B-Instruct DeepSeek-R1-Distill-Qwen-14B	Code-R1 DeepCoder	GRPO	LiveCodeBench HumanEval+
Visual Reasoning	Qwen2.5-VL-7B	EasyR1	GRPO	MathVista MathVision
Deep Analysis	Qwen2.5-7B-Base Qwen2.5-7B-Instruct DeepSeek-R1-Distill-Qwen-7B	VeRL	PPO, GRPO Reinforce++ RLOO, ReMax, DAPO	Omni-Math-Rule MATH500

3. RLVR's Effect on Reasoning Capacity Boundary

With the evaluation metrics for reasoning boundaries established, we now conduct a comprehensive evaluation of the base and RLVR models through extensive experiments. Our analysis is organized by task category, covering three representative domains: mathematics, code generation, and visual reasoning. The overall experimental setup is summarized in Table 1.

Evaluation Protocol. For sampling procedures for both base and RLVR models, we use a temperature of 0.6 and a top-p value of 0.95, allowing a maximum generation of 16,384 tokens. We also show the effect of different temperature settings in Figure 17. For evaluation of the base model, a common practice is to include few-shot examples in the prompt to guide the output (Grattafiori et al., 2024; Yang et al., 2024; Liu et al., 2024). However, to ensure a fair and unbiased comparison, we deliberately avoid using few-shot prompts for base models, eliminating any potential confounding effects on reasoning that might

be introduced by in-context examples. For evaluating both the base and RLVR models, we use the same zero-shot prompt as in RLVR training, or the default prompt provided by the benchmark, ensuring a consistent setup across both models. Interestingly, although base models often produce unformatted or non-sensical responses without few-shot guidance, we observe that with sufficient sampling, they are still capable of generating correctly formatted outputs and successfully solving complex problems. Prompt templates for training and evaluation are provided in Section D.

3.1. RLVR for Mathematical Reasoning

Models and Benchmarks. In math problems, models are required to generate a reasoning process (*i.e.*, CoT) along with the final answer. To ensure the robustness of conclusions, we experiment with multiple LLM families, primarily Qwen2.5 (7B/14B/32B base variants) (Yang et al., 2024) and additionally LLaMA-3.1-8B (Grattafiori et al., 2024). We adopt RLVR models released by SimpleRLZoo (Zeng et al., 2025), which train zero-RL models using GRPO on GSM8K and the MATH training set, with correctness reward only, excluding any format-based reward. We compare the pass@k curves of base and zero-RL models on benchmarks of varying difficulty: GSM8K (Cobbe et al., 2021), MATH500 (Hendrycks et al., 2021), Minerva (Lewkowycz et al., 2022), Olympiad (He et al., 2024), AIME24, and AMC23. Additionally, we include the RLVR model Oat-Zero-7B and DAPO-32B (Liu et al., 2025b; Yu et al., 2025). These two models are characterized by strong performance on the challenging AIME24 benchmark.

The Effect of RLVR: Increased Likelihood of Correct Samples, Decreased Coverage of Solvable Problems. As shown in Figure 2, we consistently observe a contrasting trend between small and large k values. When k is small (*e.g.*, $k = 1$, equivalent to average-case accuracy), RL-trained models outperform their base counterparts. This aligns with the common observation that RL improves performance, suggesting that RLVR makes models significantly more likely to sample correct responses. However, as k increases, with steeper curves, base models consistently catch up to and eventually surpass RL-trained models across all benchmarks, indicating their broader coverage of solvable problems. For example, on the Minerva benchmark with a 32B-sized model, the base model outperforms the RL-trained model by approximately 9% at $k = 128$, implying that it can solve around 9% more problems in the validation set.

We further examine RL models trained with Oat-Zero and DAPO. As shown in Figure 11, although the RL model initially demonstrates a strong performance, nearly 30% higher than the base model, it is eventually surpassed by the base model. Based on these results, we conclude that RLVR increases the likelihood of sampling correct responses at low k, but narrows the model's overall coverage. We further analyze the root cause of this phenomenon in Section 4.1.

CoT Case Analysis. We present the correct CoTs sampled from the base model in Figure 20 and Figure 21, manually selected from 2048 samplings for the hardest questions in AIME24. The responses from the base model tend to be long CoTs and exhibit reflective behavior, highlighting the strong reasoning ability inherent in the base model.

Validity of Chain-of-Thought. For mathematical problems, the common evaluation is based solely on the correctness of the final answer, with the risk of "hacking". To accurately reflect the reasoning ability boundary using pass@k, it is important to assess how many solved problems result from sampling genuinely correct CoTs, rather than from lucky guesses. Following (Brown et al., 2024), we manually inspect all CoTs that led to correct answers to the most challenging solvable problems in the GSM8k dataset – those with an average accuracy below 5% but above 0%. The base model answered 25 such questions, with 24 containing *at least one* correct CoT. Similarly, the RL-trained model answered 25 questions, 23 of which included *at least one* correct CoT. We also manually check the CoTs for problems in the challenging AIME24 benchmark with an average accuracy below 5%. Details can be found in Section C.2. The base model answered 7 such questions, with 5 out of 6 containing *at least one* correct CoT (excluding one ambiguous case of correctness due to skipped reasoning steps). Similarly, the RL-trained model answered 6 questions, 4 of which included *at least one* correct CoT. These results suggest that the base model can sample valid reasoning paths to solve the problems.