Temporal Sampling for Forgotten Reasoning in LLMs

Yuetai Li [*1] **Zhangchen Xu** [*1] **Fengqing Jiang** [1] **Bhaskar Ramasubramanian** [2] **Luyao Niu** [1] **Bill Yuchen Lin** [1]
Xiang Yue [3] **Radha Poovendran** [1]

Abstract

Fine-tuning large language models (LLMs) is intended to improve their reasoning capabilities, yet we uncover a counterintuitive effect: models often forget how to solve problems they previously answered correctly during training. We term this phenomenon *temporal forgetting* and show that it is widespread across model sizes, fine-tuning methods (both Reinforcement Learning and Supervised Fine-Tuning), and multiple reasoning benchmarks. To address this gap, we introduce Temporal Sampling, a simple decoding strategy that draws outputs from multiple checkpoints along the training trajectory. This approach recovers forgotten solutions and leads to substantial improvements in reasoning performance, gains from 4 to 19 points in Pass@k and consistent gains in Majority@k across several benchmarks. We further extend our method to LoRA-adapted models. By leveraging the temporal diversity inherent in training, Temporal Sampling offers a practical, compute-efficient way to surface hidden reasoning ability and rethink how we evaluate LLMs. Our code is available at: https://github.com/uw-nsl/Temporal_Forgetting

1. Introduction

Fine-tuning large language models (LLMs) is expected to improve their reasoning ability (Luo et al., 2025; DeepSeek-AI et al., 2025; Zeng et al., 2025; Muennighoff et al., 2025; NovaSky, 2025). Yet, we uncover a surprising phenomenon: *models often forget how to solve problems they previously solved correctly during fine-tuning*. We refer to this systematic behavior as *temporal forgetting*.

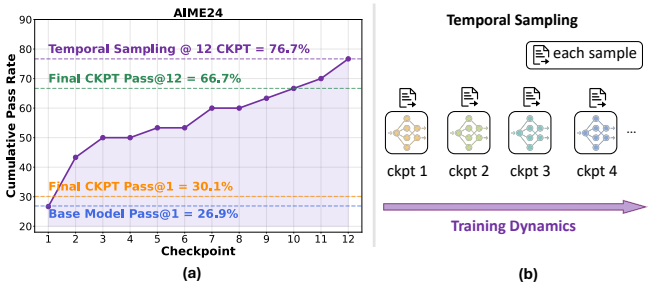

Figure 1. (a) We observed that during RL training process of Deepseek-R1-1.5B model, 76.7% of AIME problems were solved correctly at *some* intermediate checkpoint, yet only 30% remained correct in the final model. (b) We proposed Temporal Sampling: This method utilizes training dynamics as a source of answer diversity by distributing inference samples across multiple distinct checkpoints from the training trajectory, rather than relying solely on the single final checkpoint.

Temporal forgetting is not rare or model-specific. Across Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL) fine-tuning (Shao et al., 2024a; DeepSeek-AI et al., 2025; Zeng et al., 2025) of Qwen2.5 models (1.5B and 7B) on multiple reasoning benchmarks (AIME, AMC, OlympiadBench (He et al., 2024), MATH-500 (Hendrycks et al., 2021a), GPQA (Rein et al., 2024)), we find that up to 50% of final errors were once solved correctly at an earlier checkpoint. This pattern persists across different model sizes, architectures, and training approaches.

Standard metrics like Pass@k (Chen et al., 2021) and Majority@k (Wang et al., 2023b), computed on the final model, assume that checkpoint to be the model's most capable state. But our findings show that many correct reasoning paths are transient, making final-checkpoint-only evaluation a narrow and often misleading lens.

To address this gap between potential and measured performance, we introduce Temporal Sampling, a simple decoding strategy that samples completions across multiple checkpoints rather than just the final one. Temporal Sampling yields substantial improvements across diverse reasoning tasks. On benchmarks such as AIME24 and AMC, we observe gains from 4 to 19 points in Pass@k compared to final-checkpoint-only sampling, and consistent improvements in Majority@k. To make Temporal Sampling deployment-friendly, we extend it to LoRA-adapted models (Hu et al.,

[*]Equal contribution [1]University of Washington, WA, USA [2]Western Washington University, WA, USA [3]Carnegie Mellon University, PA, USA. Correspondence to: Yuetai Li <yuetai@uw.edu>, Zhangchen Xu <zxu9@uw.edu>, Radha Poovendran <rp3@uw.edu>.

2nd Workshop on Test-Time Adaptation at the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

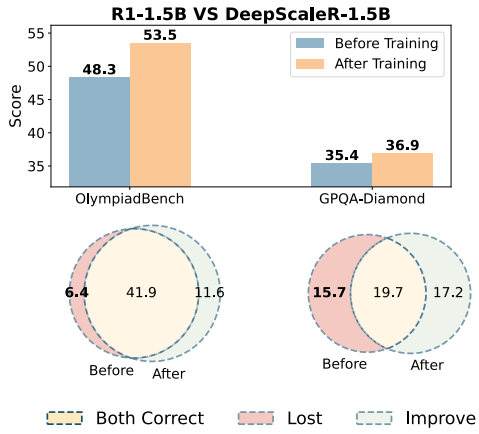

Figure 2. Overall performance score cannot tell everything. Fine-tuned models like DeepscaleR-1.5B (Luo et al., 2025) outperform the base model overall but also forget many questions the base model answered correctly.

2021).

These findings suggest that true model competence may not reside in a single parameter snapshot, but rather in the collective dynamics of training itself. Temporal Sampling offers a practical and powerful way to reclaim lost reasoning ability, challenging the standard paradigm of using only the final model checkpoint for evaluation and deployment.

2. Temporal Forgetting: Correct Answers Emerge and Vanish in Training

2.1. Overall Performance Score cannot Tell Everything

To understand how RL or SFT alters a model's ability to correctly answer reasoning problems, we investigate instances where base models succeeded on questions but failed after fine-tuning. To quantify this, we introduce the **Lost Score**:

- P_{Lost} (**Lost Score**): The percentage of questions in a benchmark that were answered correctly by the base model but incorrectly by the model after fine-tuning.

This score specifically highlights the phenomenon where a model, despite any overall performance changes after fine-tuning, loses its correctness on certain problems it previously solved correctly.

Figure 2 demonstrates that although DeepScaleR-1.5B improves GPQA performance from 35.4 to 36.9, a notable percentage of questions ($P_{Lost} = 15.7$) were correctly solved by the base model but incorrectly by the fine-tuned model.

We present a more comprehensive analysis of various SOTA models in Table 2 in Appendix D. We found that P_{Lost} could range from 6.1 to 16.0 points, with the average of 9.5 points. This implies that there are a considerable number of questions answered correctly by the base model but incorrectly after RL or SFT, in spite of the improvement of overall performance. Please also see Appendix D for the detailed experiment setup.

2.2. Temporal Forgetting

To investigate how answer correctness evolves during post-training, we conducted SFT and RL on various base models, evaluating checkpoints at different training steps. We introduce two metrics to quantify the temporal dynamics: the **Ever Correct Score** and the **Temporal Forgetting Score**:

- P_{ECS} (**Ever Correct Score**): The percentage of questions in the benchmark that were answered correctly by *at least one* checkpoint saved during RL/SFT.
- P_{TFS} (**Temporal Forgetting Score**): The percentage of questions in the benchmark that were answered correctly by *some* checkpoint during RL/SFT but were ultimately answered incorrectly by the final checkpoint. Mathematically, $P_{TFS} = P_{ECS} - P_{FT}$, where P_{FT} is the performance score of the fine-tuned model.

Experiment Setup. We performed GRPO (Shao et al., 2024b) on the Qwen2.5-7B, Qwen2.5-1.5B, and Qwen2.5-Math-7B models (Yang et al., 2024a;b). The training data consisted of 4k samples randomly selected from the DeepscaleR-40k dataset (Luo et al., 2025). Throughout the training of each model, we saved 8 checkpoints. We set the RL training parameters following (Luo et al., 2025), and detailed training script parameters can be found in Appendix E. For SFT, we utilized the same DeepscaleR-4k sampled data. We then employed QwQ-Preview-32B (Qwen Team, 2024) for rejection sampling to obtain correct responses (Dong et al., 2023), subsequently fine-tuning each model on this curated dataset. We evaluated the performance of various checkpoints from the training process on five benchmarks: AIME24, AMC, MATH-500, OlympiadBench, and GPQA-Diamond. To minimize variability caused by random fluctuations in model performance from diverse sampling, we employed greedy sampling following (Wei et al., 2022).

Results. In Figure 3 (a), we illustrate the correctness of answers to different OlympiadBench questions at various checkpoints during the RL training of Qwen2.5-7B. Figure 3 (a) demonstrates the phenomenon of **Forgetting Dynamics**: Questions exhibits alternating "Improve" and "Forget" events frequently during training, which means the model oscillates between correct and incorrect answers across checkpoints. In Figure 3 (b), we show the percentage of questions across different benchmarks that experienced the "Forget" event could achieve up to 32.3% in OlympiadBench and 52.5% in AMC.

Table 1 presents the Ever Correct Score P_{ECS} and Temporal Forgetting Score P_{TFS} of different models after RL or SFT.

Table 1. Performance of fine-tuned models (P_{FT} ↑), the Ever Correct Score (P_{ECS} ↑), and the Temporal Forgetting Score (P_{TFS} ↓) of different models after GRPO or SFT. We observed both high P_{ECS} and P_{TFS}, which implies a high percentage of questions (from 6.4% to 56.1%) are answered correctly at some checkpoint during training but are ultimately incorrect in the final checkpoint. Please see the base model performance and more benchmark results in Appendix F.

Model	OlympiadBench			MATH-500			GPQA-Diamond			Avg. P_{TFS}
	P_{FT}	P_{ECS}	P_{TFS}	P_{FT}	P_{ECS}	P_{TFS}	P_{FT}	P_{ECS}	P_{TFS}	
Qwen2.5-7B (GRPO)	39.7	58.7	19.0	73.8	89.6	15.8	33.8	74.7	40.9	25.2
Qwen2.5-7B (SFT)	40.1	55.8	15.7	69.8	86.6	16.8	25.3	81.4	56.1	29.5
Qwen2.5-1.5B (GRPO)	18.8	36.1	17.3	55.6	73.0	17.4	26.8	72.3	45.5	26.7
Qwen2.5-1.5B (SFT)	11.0	26.0	15.0	36.2	66.0	29.8	13.1	65.1	52.0	32.3
Qwen2.5-Math-7B (GRPO)	41.0	57.3	16.3	79.8	86.2	6.4	32.8	71.7	38.9	20.5
Qwen2.5-Math-7B (SFT)	43.9	62.9	19.0	76.4	90.4	14.2	30.8	79.8	49.0	27.4

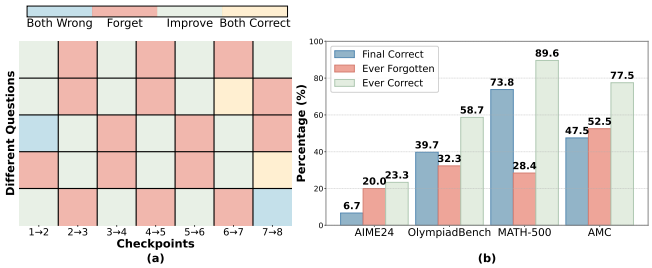

Figure 3. Forgetting dynamics of Qwen2.5-7B during RL training. (a) Answer correctness trajectories for OlympiadBench questions across training checkpoints, illustrating solutions oscillate between correct and incorrect states. "Forget" implies that an answer was correct at the previous checkpoint but incorrect at the current one, while "Improve" implies that an answer that was incorrect at the previous checkpoint but correct at the current one. (b) Percentage of questions per benchmark that are ever forgotten or ever correct at some checkpoint during RL.

We observed that a substantial number of questions were correctly answered at some checkpoint during the training process but were answered incorrectly by the final checkpoint (measured by a significantly high P_{TFS}). Surprisingly, we found that P_{TFS} ranges from 6.4% to 56.1%, with average as high as 25 points. This implies that, on average, up to 25% of the questions in a benchmark were correctly solved by the model at some checkpoint during training but were incorrect in the final output. Please see Appendix F for base model performance and more benchmark results including AIME24 and AMC.

In contrast to **Catastrophic Forgetting** (Luo et al., 2023) where overall performance drops markedly, our observed **Temporal Forgetting** focuses on changes in correctness at the *individual question level*, rather than on a collective measure, thus cannot be directly captured by the overall performance score. Temporal Forgetting emphasizes more fine-grained changes in the answer correctness shift during training dynamics, in spite of the improvement of overall performance.

3. Temporal Sampling: Scaling Inference Compute over Checkpoints

3.1. Temporal Sampling

Inspired by the observed learning and forgetting dynamics during model training, we propose **Temporal Sampling**. Temporal Sampling utilizes the evolving state of the model across different training checkpoints as a source of diversity for answer generation at inference time. Specifically, instead of relying solely on the final checkpoint, k samples are generated by allocating the sampling budget across t distinct training checkpoints according to a chosen distribution strategy.

Temporal Sampling typically selects the t most recent available checkpoints, which are then ordered from latest (e.g., the final checkpoint) to the t-th latest. While various methods can be employed to distribute the k sampling attempts among these checkpoints, this paper primarily focuses on a round-robin allocation. In this approach, sampling commences with the latest checkpoint for the first sample, the next latest for the second, and so on, cycling through the ordered sequence. This procedure defaults to sampling only from the final checkpoint when $t = 1$.

3.2. Metric $Pass@k|t$

To better measure the performance of Temporal Sampling, we introduce a new metric, $Pass@k|t$. This metric is defined as the probability of obtaining at least one correct answer when k samples are drawn from t checkpoints. Although samples may be drawn in various ways, in what follows we adopt a round-robin manner: we first give the formal definition of $Pass@k|t$ under this distribution way and then derive the unbiased estimator.

Definition. Let $r_{i,j}$ denote the $Pass@1$ rate (i.e., the probability of correctness with a single sample) for the j-th

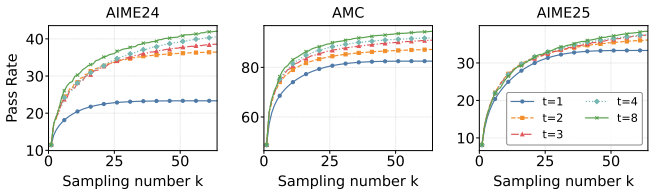

Figure 4. $Pass@k$ for different numbers of checkpoints t on the AIME2024, AMC, and AIME2025 benchmarks when using Temporal Sampling. The case $t = 1$ represents the baseline of standard $Pass@k$ sampling on the final checkpoint. Our proposed Temporal Sampling with $t = 8$ outperforms the baseline by more than 19, 13, and 4 percentage points on AIME2024, AMC, and AIME2025, respectively, when sampling 64 responses.

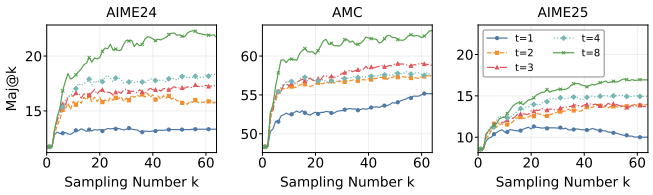

Figure 5. Maj@k (Majority voting) for different numbers of checkpoints t on the AIME2024, AMC, and AIME2025 benchmarks using Temporal Sampling. The case $t = 1$ represents the baseline of standard majority voting sampling on the final checkpoint. Our proposed Temporal Sampling with $t = 8$ checkpoints outperforms the baseline by more than 8, 7, and 7 percentage points on AIME2024, AMC, and AIME2025, respectively, when sampling 64 responses.

checkpoint on the i-th problem. We define

$$Pass@k|t = \mathop{\mathbb{E}}_{i}\left\{1 - \prod_{j=1}^{t}(1 - r_{i,j})^{k_j}\right\}$$

where $\sum_j k_j = k$ and $\{k_j\}$ is the *Balanced Integer Partition* of k on t (Andrews & Eriksson, 2004):

$$k_j = \begin{cases} \lfloor k/t \rfloor + 1 & \text{if } j \leq (k \pmod t) \\ \lfloor k/t \rfloor & \text{if } j > (k \pmod t) \end{cases}$$

Note that if $t = 1$, this reduces to the standard definition of $Pass@k$ (Chen et al., 2021).

Unbiased Estimation. We provide an unbiased estimator of the proposed Pass@$@k|t$. Please see Appendix C for more details.

3.3. Experiment Setup

We utilized GRPO to fine-tune the Qwen-7B-Base model on the DeepScaleR-4k dataset, following the training settings in (Luo et al., 2025). We saved 8 checkpoints during RL, which constituted the checkpoint pool for our Temporal Sampling. As baselines, we considered the standard $Pass@k$ (Chen et al., 2021) and Maj@k (self-consistency, also known as majority voting) (Wang et al., 2023a). For Maj@k, we followed the Majority Voting (Wang et al., 2023a) by generating k samples and selecting the most frequent answer as the final model output. We denote our Temporal Sampling variants as $Pass@k|t$ and Maj@$k|t$. When $t = 1$, $Pass@k|t$ and $Maj@k|t$ are equivalent to the baseline settings that samples only on the final checkpoint.

3.4. Temporal Sampling Achieves Higher Sampling Performance

Figure 4 demonstrates that Temporal Sampling achieves higher sampling performance compared to the baseline of sampling only on the final checkpoint, under identical computational budgets. These advantages are consistently observed across the AIME2024, AIME2025, and AMC benchmarks. For instance $Pass@k|8$ results in a pass rate that is over 19 percentage points higher than that of sampling only on the final checkpoint on AIME24 when $k = 64$. Please see Appendix G.1 for more experiment results on different models.

3.5. Temporal Sampling Enhances Performance of Inference-Time Scaling

Figure 5 demonstrates that Temporal Sampling markedly enhances the performance of majority voting (measured by $Maj@k|t$). Across the AIME2024, AIME2025, and AMC benchmarks, employing a greater number of checkpoints (t) within the Temporal Sampling framework leads to improved accuracy compared to the baseline Maj@k only sampling on the final checkpoint under identical computational budgets. Specifically, at $k = 64$, $Maj@k|8$ achieves an accuracy exceeding 21, substantially outperforming the 13% accuracy of the baseline.

We present further experiments regarding Best-of-N sampling in Appendix G.2.

3.6. Temporal Sampling with LoRA Fine-tuning

To save the storage cost associated with multiple model checkpoints, we investigated the use of Low-Rank Adaptation (LoRA) for Fine-Tuning, where checkpoints generated only store the low-rank adapter weights, smaller than full parameter fine-tuning. We demonstrate that LoRA-based Temporal Sampling improves both Pass@k and Majority@k than sampling only on the final checkpoint. Please see more details in Appendix G.3.

4. Conclusion

In this paper, we observed the phenomenon of Temporal Forgetting: models often forget how to solve problems they previously solved correctly during fine-tuning. Our analysis of training trajectories revealed that many correct solutions

emerge transiently during training but are absent in the final model. Inspired by the training dynamics, we propose Temporal Sampling, a simple inference-time method that samples from multiple training checkpoints to recover forgotten solutions. This approach consistently improves reasoning performance by 4-19 points in Pass@k across benchmarks and can be efficiently implemented using LoRA. These findings suggest that true model competence may not reside in a single parameter snapshot, but rather in the collective dynamics of training itself. Temporal Sampling offers a practical and powerful way to reclaim lost reasoning ability, challenging the standard paradigm of using only the final model checkpoint for evaluation and deployment.

References

Andrews, G. E. and Eriksson, K. *Integer partitions*. Cambridge University Press, 2004.

Bae, J., Lin, W., Lorraine, J., and Grosse, R. Training data attribution via approximate unrolled differentiation, 2024. URL https://arxiv.org/abs/2405.12186.

Beeching, E., Tunstall, L., and Rush, S. Scaling test-time compute with open models, 2024. URL https://huggingface.co/spaces/HuggingFaceH4/blogpost-scaling-test-time-compute.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. D. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., Xue, B., Wang, B., Wu, B., Feng, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., Dai, D., Chen, D., Ji, D., Li, E., Lin, F., Dai, F., Luo, F., Hao, G., Chen, G., Li, G., Zhang, H., Bao, H., Xu, H., Wang, H., Ding, H., Xin, H., Gao, H., Qu, H., Li, H., Guo, J., Li, J., Wang, J., Chen, J., Yuan, J., Qiu, J., Li, J., Cai, J. L., Ni, J., Liang, J., Chen, J., Dong, K., Hu, K., Gao, K., Guan, K., Huang, K., Yu, K., Wang, L., Zhang, L., Zhao, L., Wang, L., Zhang, L., Xu, L., Xia, L., Zhang, M., Zhang, M., Tang, M., Li, M., Wang, M., Li, M., Tian, N., Huang, P., Zhang, P., Wang, Q., Chen, Q., Du, Q., Ge, R., Zhang, R., Pan, R., Wang, R., Chen, R. J., Jin, R. L., Chen, R., Lu, S., Zhou, S., Chen, S., Ye, S., Wang, S., Yu, S., Zhou, S., Pan, S., Li, S. S., Zhou, S., Wu, S., Ye, S., Yun, T., Pei, T., Sun, T., Wang, T., Zeng, W., Zhao, W., Liu, W., Liang, W., Gao, W., Yu, W., Zhang, W., Xiao, W. L., An, W., Liu, X., Wang, X., Chen, X., Nie, X., Cheng, X., Liu, X., Xie, X., Liu, X., Yang, X., Li, X., Su, X., Lin, X., Li, X. Q., Jin, X., Shen, X., Chen, X., Sun, X., Wang, X., Song, X., Zhou, X., Wang, X., Shan, X., Li, Y. K., Wang, Y. Q., Wei, Y. X., Zhang, Y., Xu, Y., Li, Y., Zhao, Y., Sun, Y., Wang, Y., Yu, Y., Zhang, Y., Shi, Y., Xiong, Y., He, Y., Piao, Y., Wang, Y., Tan, Y., Ma, Y., Liu, Y., Guo, Y., Ou, Y., Wang, Y., Gong, Y., Zou, Y., He, Y., Xiong, Y., Luo, Y., You, Y., Liu, Y., Zhou, Y., Zhu, Y. X., Xu, Y., Huang, Y., Li, Y., Zheng, Y., Zhu, Y., Ma, Y., Tang, Y., Zha, Y., Yan, Y., Ren, Z. Z., Ren, Z., Sha, Z., Fu, Z., Xu, Z., Xie, Z., Zhang, Z., Hao, Z., Ma, Z., Yan, Z., Wu, Z., Gu, Z., Zhu, Z., Liu, Z., Li, Z., Xie, Z., Song, Z., Pan, Z., Huang, Z., Xu, Z., Zhang, Z., and Zhang, Z. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Dong, H., Xiong, W., Goyal, D., Zhang, Y., Chow, W., Pan, R., Diao, S., Zhang, J., Shum, K., and Zhang, T. Raft: Reward ranked finetuning for generative foundation model alignment, 2023. URL https://arxiv.org/abs/2304.06767.

Face, H. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL https://github.com/huggingface/open-r1.

Feng, X., Wan, Z., Wen, M., McAleer, S. M., Wen, Y., Zhang, W., and Wang, J. Alphazero-like tree-search can guide large language model decoding and training, 2023.

He, C., Luo, R., Bai, Y., Hu, S., Thai, Z. L., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., Liu, J., Qi, L., Liu, Z., and Sun, M. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems, 2024. URL https://arxiv.org/abs/2402.14008.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. In Vanschoren, J. and Yeung, S. (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021a.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset, 2021b. URL https://arxiv.org/abs/2103.03874.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Huang, S. C. and Ahmadian, A. Putting rl back in rlhf. https://huggingface.co/blog/putting_

rl_back_in_rlhf_with_rloo, June 12 2024. Hugging Face Blog.

Internet, I. Ii-thought : A large-scale, high-quality reasoning dataset, 2025.

Kang, J., Li, X. Z., Chen, X., Kazemi, A., Sun, Q., Chen, B., Li, D., He, X., He, Q., Wen, F., et al. MindStar: Enhancing math reasoning in pre-trained llms at inference time. *arXiv preprint arXiv:2405.16265*, 2024.

Khanov, M., Burapacheep, J., and Li, Y. ARGS: Alignment as reward-guided search. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=shgx0eqdw6.

Kimi Team. Kimi k1.5: Scaling reinforcement learning with llms, 2025.

Kumar, A., Raghunathan, A., Jones, R., Ma, T., and Liang, P. Fine-tuning can distort pretrained features and underperform out-of-distribution, 2022. URL https://arxiv.org/abs/2202.10054.

Lai, Y., Zhong, J., Li, M., Zhao, S., and Yang, X. Med-r1: Reinforcement learning for generalizable medical reasoning in vision-language models, 2025. URL https://arxiv.org/abs/2503.13939.

Lambert, N., Morrison, J., Pyatkin, V., Huang, S., Ivison, H., Brahman, F., Miranda, L. J. V., Liu, A., Dziri, N., Lyu, S., Gu, Y., Malik, S., Graf, V., Hwang, J. D., Yang, J., Bras, R. L., Tafjord, O., Wilhelm, C., Soldaini, L., Smith, N. A., Wang, Y., Dasigi, P., and Hajishirzi, H. Tulu 3: Pushing frontiers in open language model post-training, 2024.

Luo, M., Tan, S., Wong, J., Shi, X., Tang, W. Y., Roongta, M., Cai, C., Luo, J., Li, L. E., Popa, R. A., and Stoica, I. 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-19681902c1468005bed8ca303, 2025. Notion Blog.

Luo, Y., Yang, Z., Meng, F., Li, Y., Zhou, J., and Zhang, Y. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *arXiv preprint arXiv:2308.08747*, 2023.

Manvi, R., Singh, A., and Ermon, S. Adaptive inference-time compute: Llms can predict if they can do better, even mid-generation. *arXiv preprint arXiv:2410.02725*, 2024.

Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. s1: Simple test-time scaling, 2025. URL https://arxiv.org/abs/2501.19393.

NovaSky. Sky-T1: Train your own o1 preview model within $450, 2025. URL https://novasky-ai.github.io/posts/sky-t1. Accessed: 2025-01-09.

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

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback, 2022.

Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown, 2024. URL https://qwenlm.github.io/blog/qwq-32b-preview/.

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., and Finn, C. Direct preference optimization: Your language model is secretly a reward model, 2024. URL https://arxiv.org/abs/2305.18290.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

Ren, Y. and Sutherland, D. J. Learning dynamics of llm finetuning, 2025. URL https://arxiv.org/abs/2407.10490.

Ren, Y., Guo, S., Bae, W., and Sutherland, D. J. How to prepare your task head for finetuning, 2023. URL https://arxiv.org/abs/2302.05779.

Sardana, N., Portes, J., Doubov, S., and Frankle, J. Beyond chinchilla-optimal: Accounting for inference in language model scaling laws. In *International Conference on Machine Learning (ICML)*, volume 235, pp. 43445–43460, 2024.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024a.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M., Li, Y., Wu, Y., and Guo, D. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024b.

Sheng, G., Zhang, C., Ye, Z., Wu, X., Zhang, W., Zhang, R., Peng, Y., Lin, H., and Wu, C. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Snell, C., Lee, J., Xu, K., and Kumar, A. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Team, O. Open Thoughts. https://open-thoughts.ai, January 2025a.

Team, R. S. Still-3-1.5b-preview: Enhancing slow thinking abilities of small models through reinforcement learning. 2025b. URL https://github.com/RUCAIBox/Slow_Thinking_with_LLMs.

Wan, Z., Feng, X., Wen, M., Mcaleer, S. M., Wen, Y., Zhang, W., and Wang, J. AlphaZero-like tree-search can guide large language model decoding and training. In *International Conference on Machine Learning (ICML)*, volume 235, pp. 49890–49920, 2024.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models, 2023a. URL https://arxiv.org/abs/2203.11171.

Wang, X., Wei, J., Schuurmans, D., Le, Q. V., Chi, E. H., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023b. URL https://openreview.net/forum?id=1PL1NIMMrw.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Wei, Y., Duchenne, O., Copet, J., Carbonneaux, Q., Zhang, L., Fried, D., Synnaeve, G., Singh, R., and Wang, S. I. Swe-rl: Advancing llm reasoning via reinforcement learning on open software evolution. *arXiv preprint arXiv:2502.18449*, 2025.

Wu, Y., Sun, Z., Li, S., Welleck, S., and Yang, Y. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*, 2024.

Xie, Y., Kawaguchi, K., Zhao, Y., Zhao, J. X., Kan, M.-Y., He, J., and Xie, M. Self-evaluation guided beam search for reasoning. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 41618–41650. Curran Associates, Inc., 2023.

Xin, H., Guo, D., Shao, Z., Ren, Z., Zhu, Q., Liu, B., Ruan, C., Li, W., and Liang, X. Deepseek-prover: Advancing theorem proving in llms through large-scale synthetic data. *arXiv preprint arXiv:2405.14333*, 2024.

Xiong, W., Yao, J., Xu, Y., Pang, B., Wang, L., Sahoo, D., Li, J., Jiang, N., Zhang, T., Xiong, C., and Dong, H. A minimalist approach to llm reasoning: from rejection sampling to reinforce, 2025. URL https://arxiv.org/abs/2504.11343.

Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., and Qiu, Z. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

Yang, A., Zhang, B., Hui, B., Gao, B., Yu, B., Li, C., Liu, D., Tu, J., Zhou, J., Lin, J., Lu, K., Xue, M., Lin, R., Liu, T., Ren, X., and Zhang, Z. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024b.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., and Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 11809–11822, 2023.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Fan, T., Liu, G., Liu, L., Liu, X., Lin, H., Lin, Z., Ma, B., Sheng, G., Tong, Y., Zhang, C., Zhang, M., Zhang, W., Zhu, H., Zhu, J., Chen, J., Chen, J., Wang, C., Yu, H., Dai, W., Song, Y., Wei, X., Zhou, H., Liu, J., Ma, W.-Y., Zhang, Y.-Q., Yan, L., Qiao, M., Wu, Y., and Wang, M. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Zeng, W., Huang, Y., Liu, W., He, K., Liu, Q., Ma, Z., and He, J. 7b model and 8k examples: Emerging reasoning with reinforcement learning is both effective and efficient. https://hkust-nlp.notion.site/simplerl-reason, 2025. Notion Blog.

Zhang, Z., Zheng, C., Wu, Y., Zhang, B., Lin, R., Yu, B., Liu, D., Zhou, J., and Lin, J. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*, 2025.

Zheng, Y., Zhang, R., Zhang, J., Ye, Y., Luo, Z., Feng, Z., and Ma, Y. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

A. Related Work

Reinforcement learning for LLM. Reinforcement Learning (RL) has rapidly become a cornerstone for extending the capabilities of LLMs across various applications. Although it was first employed to align model behavior with human preferences through approaches like Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022), its role now encompasses reasoning on complex tasks (Kimi Team, 2025; DeepSeek-AI, 2025; Lambert et al., 2024). For example, DeepSeek-R1 applied RL directly to a base "zero" LLM (DeepSeek-AI, 2025), and Kimi K1.5 augmented this framework with multimodal reasoning and verbosity control (Kimi Team, 2025). In particular, Reinforcement Learning has gained traction in areas such as mathematics and programming, where reward signals can be defined by clear, rule-based criteria like answer matching (Lambert et al., 2024; Shao et al., 2024b; Chen et al., 2021; DeepSeek-AI, 2025; Feng et al., 2023; Snell et al., 2024; Xie et al., 2023; Wan et al., 2024). Advances in optimization, such as specialized PPO variants (e.g., VinePPO (Feng et al., 2023)) and stabilized GRPO algorithms (e.g., DAPO (Yu et al., 2025)), have simplified reward design, making RL more practical. Our work shifts focus from static performance gains of RL to the evolution of answer correctness over the procedure of RL training. We harness these temporal fluctuations as the diversity source to increase inference-time performance.

Inference Time Scaling. Expanding the computational budget available during inference has become a powerful lever for squeezing extra performance out of large language models, giving rise to an ever-growing family of test-time scaling (TTS) techniques (OpenAI, 2024). The field has seen a variety of approaches to leverage this. Established techniques include sampling-driven methods like majority voting (Wang et al., 2023b) or best-of-N (Sardana et al., 2024), which generate many candidate answers and select the most persuasive one. More intricate are search-based algorithms such as Tree-of-Thoughts (ToT) explorations (Yao et al., 2023) and Monte-Carlo tree search (MCTS) (Xie et al., 2023; Khanov et al., 2024; Wan et al., 2024). Such approaches often build upon the development of sophisticated verifiers and may integrate process-based reward signals directly into search methods (Kang et al., 2024; Wu et al., 2024; Snell et al., 2024). To further enhance efficiency and adaptiveness, other techniques include self-evaluation mechanisms for judicious compute allocation (Manvi et al., 2024) and diversity-aware search tactics, sometimes referred to as Test-Time Scaling (TTS) with diversity, to reduce redundant sampling and explore a wider solution space (Beeching et al., 2024).

Learning Dynamics. Learning dynamics analyze model behavior during training, such as explaining "aha moments" (DeepSeek-AI, 2025), and challenges in fine-tuning generalization (e.g., (Kumar et al., 2022; Ren et al., 2023)). These works focus on the training process itself and offer novel perspectives on how models learn and develop capabilities. Other research analyzes the step-wise decomposition of how influence accumulates among different potential responses for both instruction and preference tuning in LLMs (Ren & Sutherland, 2025). This detailed analytical framework, offering hypothetical explanations for why specific types of hallucination are strengthened post-finetuning. From the data perspective, Training Data Attribution (TDA) (Bae et al., 2024) identifies influential training examples to explain model predictions. Orthogonal to these works, we empirically investigate the dynamic fluctuations in answer correctness across diverse reasoning tasks, and harness the learning dynamics as a source of answer diversity to widen the sampling space and performance.

B. Limitations and Broader Impacts

Our investigation into the **Temporal Forgetting** phenomenon has primarily concentrated on mathematical reasoning tasks. We have not yet extended our analysis to other potentially relevant domains where similar patterns might emerge, such as automated theorem proving (Xin et al., 2024), healthcare applications (Lai et al., 2025), or code generation (Wei et al., 2025). The experimental foundation of our work focuses on GRPO (Shao et al., 2024b) and SFT frameworks. While we believe our findings can generalize to other training methodologies, including on-policy approaches like PPO (Schulman et al., 2017), RLOO (Huang & Ahmadian, 2024), and DAPO (Yu et al., 2025), as well as off-policy techniques such as DPO (Rafailov et al., 2024), RAFT (Dong et al., 2023), and Reinforce-Rej (Xiong et al., 2025) that rely on rejection sampling. we have not empirically validated this hypothesis.

When implementing Temporal Sampling, we focus on round-robin allocation strategies for distributing the k sampling attempts across t checkpoints. Alternative distribution approaches represent a promising avenue that we reserve for subsequent research.

Broader Impacts. Through our research, we have uncovered the temporal forgetting phenomenon and developed temporal sampling as an effective method to enhance inference-time sampling performance in mathematical reasoning. We have not

identified negative societal implications associated with this work.

C. Unbiased Estimation of Pass@$k|t$

We provide an unbiased estimator for Pass@$k|t$ and show the proof of the unbiased nature in this section.

The Pass@$k|t$ metric measures the probability of obtaining at least one correct answer when samples are drawn from multiple checkpoints. The following theorem establishes the statistical validity of our evaluation framework, ensuring that our empirical measurements accurately reflect the true performance of **Temporal Sampling** across different checkpoints.

Theorem 1. *Denote $r_{i,j}$ as the Pass@1 rate for the j-th checkpoint on problem i, $C_{i,j}$ as the number of correct samples among N candidates for problem i from checkpoint j. Let*

$$P_i = 1 - \prod_{j=1}^{t} (1 - r_{i,j})^{k_j}$$

denote the probability of obtaining at least one correct answer when k samples are drawn from t checkpoints for problem i, (i.e., Pass@$k|t$), where k_j is determined by the balanced integer partition of k on t:

$$k_j = \begin{cases} \lfloor k/t \rfloor + 1 & \text{if } j \leq (k \pmod t) \\ \lfloor k/t \rfloor & \text{if } j > (k \pmod t) \end{cases}$$

We have

$$\hat{P}_i = 1 - \prod_{j=1}^{t} \left(\frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}} \right)$$

is an unbiased estimator of P_i, i.e., $\mathbb{E}[\hat{P}_i] = P_i$.

Proof. For a single checkpoint j on problem i, we consider the probability of obtaining no correct solutions when sampling k_j solutions without replacement from N total samples. Given that $C_{i,j}$ of these N samples are correct, this probability follows the hypergeometric distribution:

$$P(X_{i,j} = 0) = \frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}}$$

For Pass@$k|t$, we succeed if at least one sample across all checkpoints is correct. The probability of failure (no correct solutions from any checkpoint) is:

$$P(\text{failure}) = \prod_{j=1}^{t} P(X_{i,j} = 0) = \prod_{j=1}^{t} \frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}}$$

Thus, our estimator for the success probability is:

$$\hat{P}_i = 1 - \prod_{j=1}^{t} \frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}}$$

To prove this estimator is unbiased, we need to show that $\mathbb{E}[\hat{P}_i] = P_i$. We first prove that:

$$\mathbb{E}\left[\frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}} \right] = (1 - r_{i,j})^{k_j}$$

Since $C_{i,j}$ follows a binomial distribution $B(N, r_{i,j})$, we have:

$$\mathbb{E}\left[\frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}} \right] = \sum_{c=0}^{N} \frac{\binom{N-c}{k_j}}{\binom{N}{k_j}} \cdot \binom{N}{c} r_{i,j}^{c} (1 - r_{i,j})^{N-c} \tag{1}$$

We can simplify the coefficient:

$$\frac{\binom{N-c}{k_j}}{\binom{N}{k_j}} \cdot \binom{N}{c} = \frac{(N-c)!}{k_j!(N-c-k_j)!} \cdot \frac{k_j!(N-k_j)!}{N!} \cdot \frac{N!}{c!(N-c)!} \tag{2}$$

$$= \binom{N-k_j}{c} \tag{3}$$

Substituting this back:

$$\mathbb{E}\left[\frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}}\right] = \sum_{c=0}^{N-k_j} \binom{N-k_j}{c} r_{i,j}^c (1-r_{i,j})^{N-c} \tag{4}$$

$$= (1-r_{i,j})^{k_j} \sum_{c=0}^{N-k_j} \binom{N-k_j}{c} r_{i,j}^c (1-r_{i,j})^{N-k_j-c} \tag{5}$$

The summation represents the binomial expansion of $(r_{i,j} + (1-r_{i,j}))^{N-k_j} = 1^{N-k_j} = 1$, yielding:

$$\mathbb{E}\left[\frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}}\right] = (1-r_{i,j})^{k_j} \tag{6}$$

Since the samples from different checkpoints are independent, we have:

$$\mathbb{E}\left[\prod_{j=1}^{t} \frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}}\right] = \prod_{j=1}^{t} E\left[\frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}}\right] = \prod_{j=1}^{t} (1-r_{i,j})^{k_j} \tag{7}$$

Therefore:

$$\mathbb{E}[\hat{P}_i] = 1 - E\left[\prod_{j=1}^{t} \frac{\binom{N-C_{i,j}}{k_j}}{\binom{N}{k_j}}\right] = 1 - \prod_{j=1}^{t} (1-r_{i,j})^{k_j} = P_i \tag{8}$$

This proves that $\hat{P}_i$ is an unbiased estimator for Pass@$k|t$. □

D. Overall Performance Score cannot Tell Everything

To understand how RL or SFT alters a model's ability to correctly answer reasoning problems, we investigate instances where base models succeeded on questions but failed after fine-tuning. To quantify this, we introduce the **Lost Score**:

- P_{Lost} (**Lost Score**): The percentage of questions in a benchmark that were answered correctly by the base model but incorrectly by the model after fine-tuning.

This score specifically highlights the phenomenon where a model, despite any overall performance changes after fine-tuning, loses its correctness on certain problems it previously solved correctly. Note that overall performance scores cannot capture the statistical pattern reflected by P_{Lost}.

Experiment Setup. We consider various existing SOTA model such as DeepScaleR-1.5B (Luo et al., 2025), OpenR1-7B (Face, 2025) and S1-32B (Muennighoff et al., 2025). We calculate the overall performance of various SOTA models after fine-tuning (denoted P_{FT}), the performance of their corresponding base model (denoted P_{Base}), and our proposed Lost Score (P_{Lost}). These evaluations were conducted on the OlympiadBench (He et al., 2024), MATH-500 (Hendrycks et al., 2021b), and GPQA (Rein et al., 2024) benchmarks. We excluded AIME2024 and AMC2023 from this particular analysis because the number of questions available in these datasets was insufficient for a meaningful comparison. To minimize variability arising from different sampling methods during evaluation, we employ greedy sampling following (Wei et al., 2022).

Table 2. Performance of the base model ($P_{\text{Base}} \uparrow$), the fine-tuned model ($P_{\text{FT}} \uparrow$) and the Lost Score ($P_{Lost} \downarrow$) for different SOTA models. We observed that in spite of the improvement of overall performance, the average P_{Lost} ranges from 6.1 to 16.0, which implies a high percentage of questions answered correctly by the base model is answered incorrectly after RL or SFT.

Model	OlympiadBench			MATH-500			GPQA-Diamond			Avg. P_{Lost}
	P_{Base}	P_{FT}	P_{Lost}	P_{Base}	P_{FT}	P_{Lost}	P_{Base}	P_{FT}	P_{Lost}	
DeepScaleR-1.5B (Luo et al., 2025)	48.3	53.5	6.4	82.0	89.8	2.4	35.4	36.9	15.7	8.2
Still-1.5B (Team, 2025b)	48.3	48.4	8.6	82.0	83.8	5.0	35.4	34.8	17.2	10.3
S1.1-1.5B (Muennighoff et al., 2025)	18.7	11.7	11.1	46.2	37.6	19.2	23.2	16.2	17.7	16.0
II-thought-1.5B (Internet, 2025)	48.3	58.4	5.3	82.0	88.0	3.4	35.4	34.3	16.7	8.5
S1.1-3B (Muennighoff et al., 2025)	29.8	24.7	12.4	65.0	64.8	10.2	32.8	30.3	18.7	13.8
SmallThinker-3B	29.8	38.2	6.2	65.0	69.2	9.8	32.8	28.3	21.7	12.6
S1.1-7B (Muennighoff et al., 2025)	40.4	42.2	10.5	76.0	76.8	7.8	32.8	41.4	15.2	11.2
OpenR1-Qwen-7B (Face, 2025)	42.5	56.6	9.2	83.0	89.8	3.8	29.8	41.9	12.1	8.4
OpenThinker-7B (Team, 2025a)	40.4	48.7	8.1	76.0	85.0	4.2	32.8	43.9	13.6	8.6
S1-32B (Muennighoff et al., 2025)	49.8	60.1	4.3	81.6	89.6	3.2	43.9	55.1	13.1	6.9
Sky-T1-32B-Preview (NovaSky, 2025)	49.8	58.4	4.6	81.6	88.2	3.0	43.9	53.0	11.1	6.2
Bespoke-Stratos-32B	49.8	54.2	7.1	81.6	89.2	3.0	43.9	57.6	8.1	6.1
OpenThinker-32B (Team, 2025a)	49.8	61.2	8.0	81.6	91.4	2.8	43.9	59.1	11.1	7.3

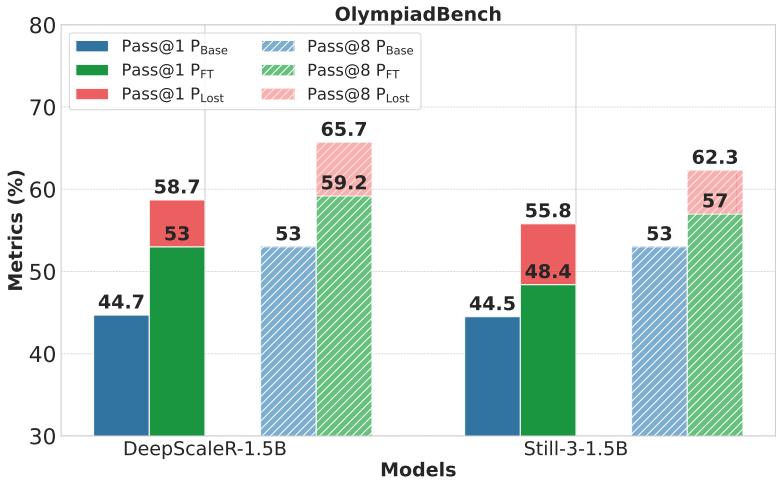

Figure 6. Performance of the base model ($P_{\text{Base}} \uparrow$), the fine-tuned model ($P_{\text{FT}} \uparrow$) and the Lost Score ($P_{Lost} \downarrow$) for Pass@1 sampling and Pass@8 sampling. Fine-tuned models like DeepscaleR-1.5B (Luo et al., 2025) and Still-3-1.5B (Face, 2025) outperform the base model overall but also forget many questions the base model answered correctly.

Results. In Table 2, we present a comprehensive analysis of various SOTA models. We found that P_{Lost} could range from 6.1 to 16.0 points, with the average of 9.5 points. This implies that there are a considerable number of questions answered correctly by the base model but incorrectly after RL or SFT, in spite of the improvement of overall performance.

Figure 6 illustrates the performance comparison between base models and fine-tuned models using both Pass@1 and Pass@8 sampling on the OlympiadBench dataset. The figure shows that while fine-tuned models like DeepscaleR-1.5B and Still-3-1.5B achieve higher overall performance than their base models ($P_{\text{FT}} > P_{\text{Base}}$), they also exhibit the temporal forgetting phenomenon with substantial Lost Scores (P_{Lost}) for both Pass@1 sampling and Pass@8 sampling.

E. Experiment Setup of Temporal Forgetting

E.1. GRPO

We follow (Luo et al., 2025) and use the following hyper-parameters detailed in Table 3 for Zero RL training. We perform experiments on eight A100 GPUs. The model is trained using VERL (Sheng et al., 2024).

Table 3. This table shows the hyper-parameters for zero RL training.

Hyper-parameter	Value
Learning Rate	1×10^{-6}
Number of Epochs	9
Number of Devices	8
Rollout Batch Size	128
PPO Mini Batch Size	64
Max Prompt Length	1024
Max Response Length	3072 (QWEN2.5-MATH-7B), 8192 (OTHERS)
KL Coefficient	0.001
Rollout Engine	VLLM (V0.8.2)
Optimizer	`Adamw`
Learning Rate Scheduler	`cosine`
Warmup Ratio	0.1

E.2. Supervised Fine-tuning

Our model SFT is conducted using LLaMA-Factory (Zheng et al., 2024), on a server with four NVIDIA A100-SXM4-80GB GPUs. We follow (NovaSky, 2025) for the training parameters. Table 4 lists hyper-parameters for full parameter supervised fine-tuning.

Table 4. This table shows the hyper-parameters for full parameter supervised fine-tuning.

Hyper-parameter	Value
Learning Rate	1×10^{-5}
Number of Epochs	3
Number of Devices	4
Per-device Batch Size	1
Optimizer	`Adamw`
Learning Rate Scheduler	`cosine`
Max Sequence Length	16384

E.3. LoRA Fine-tuning Setup

Our model LoRA fine-tuning (Hu et al., 2021) is conducted using LLaMA-Factory (Zheng et al., 2024), on a server with four NVIDIA A100-SXM4-80GB GPUs. We follow (NovaSky, 2025) for the training parameters. Table 5 lists hyper-parameters for LoRA fine-tuning.

F. More Results of Temporal Forgetting

Table F presents detailed performance metrics for different fine-tuned models evaluated specifically on AIME24 and AMC benchmarks. The table shows the base model performance (P_{Base}), fine-tuned model performance (P_{FT}), Ever Correct Score (P_{ECS}), and Temporal Forgetting Score (P_{TFS}) across various models with both GRPO and SFT training methods. Notably, models exhibit significant temporal forgetting, with P_{TFS} values ranging from 6.7% to 30%, which implies that many questions solved correctly at some point during training were ultimately answered incorrectly in the final checkpoint.

Table F complements Table 1 by providing a more comprehensive view of base model (P_{Base}) and fine-tuned model (P_{FT})

Table 5. This table shows the hyper-parameters for LoRA fine-tuning.

Hyper-parameter	Value
Learning Rate	1×10^{-4}
Number of Epochs	3
Number of Devices	4
Per-device Batch Size	1
LoRA Target	`full`
Learning Rate Scheduler	`cosine`
Warmup Ratio	0.03
Max Sequence Length	16384

Model	AMC				AIME24			
	P_{Base}	P_{FT}	P_{ECS}	P_{TFS}	P_{Base}	P_{FT}	P_{ECS}	P_{TFS}
Qwen2.5-7B (GRPO)	32.5	47.5	77.5	30.0	6.7	6.7	23.4	16.7
Qwen2.5-7B (SFT)	32.5	52.5	75.0	22.5	6.7	10.0	20.0	10.0
Qwen2.5-1.5B (GRPO)	0.0	30.0	45.0	15.0	0.0	3.3	10.0	6.7
Qwen2.5-1.5B (SFT)	0.0	15.0	35.0	20.0	0.0	0.0	6.7	6.7
Qwen2.5-Math-7B (GRPO)	32.5	72.5	82.5	10.0	13.3	16.7	40.0	23.3
Qwen2.5-Math-7B (SFT)	32.5	50.0	75.0	25.0	13.3	20.0	40.0	20.0

Table 6. Performance of fine-tuned models ($P_{FT} \uparrow$), the Ever Correct Score ($P_{ECS} \uparrow$), and the Temporal Forgetting Score ($P_{TFS} \downarrow$) of different fine-tuned models evaluated on AIME24 and AMC. We observed both high P_{ECS} and P_{TFS} in spite of the improving overall performance, which implies a high percentage of questions (from 6.7% to 30%) are answered correctly at some checkpoint during training but are ultimately incorrect in the final checkpoint.

performance across all five mathematical benchmar.

G. More results of Temporal Sampling

G.1. More Results of Temporal Sampling for Different Models

Figure 7 presents a comprehensive evaluation of Temporal Sampling across different models (Qwen2.5-7B, Qwen2.5-1.5B, and Qwen2.5-Math-7B) and benchmarks. The results consistently show that Temporal Sampling with multiple checkpoints ($t = 2$, $t = 4$, $t = 8$) outperforms the baseline ($t = 1$) across different model sizes and benchmarks.

G.2. Temporal Sampling for Best-of-N

Figure 8 demonstrates the effectiveness of Temporal Sampling when combined with Best-of-N (BoN) decoding on the AIME2024, AMC, and AIME2025 benchmarks. Using Qwen2.5-Math-PRM-72B (Zhang et al., 2025) as the process reward model, answers with the highest reward were selected as the final output. The results clearly show that Temporal Sampling with $t = 8$ checkpoints significantly outperforms the baseline ($t = 1$), achieving improvements of more than 7, 8, and 1 percentage points across the three benchmarks when sampling $k = 64$ responses. Figure 9 presents additional evidence for the effectiveness of Temporal Sampling with Best-of-N decoding when using the smaller Qwen2.5-Math-PRM-7B (Zhang et al., 2025) as the process reward model. This highlights the value of leveraging multiple training checkpoints for enhancing reward-based selection methods.

G.3. Temporal Sampling with LoRA Fine-tuning

A key consideration for the practical application of Temporal Sampling is the storage cost associated with saving multiple model checkpoints. To address this, we investigated the use of Low-Rank Adaptation (LoRA) for Fine-Tuning, where checkpoints generated only store the low-rank adapter weights, smaller than full parameter fine-tuning. In our experiments, we use LoRA SFT Qwen2.5-7B model on the DeepscaleR-4k dataset used in Section 2.2. Please see Appendix E.3 for

Model	Olympiad		MATH-500		GPQA		AMC		AIME	
	P_{Base}	P_{FT}	P_{Base}	P_{FT}	P_{Base}	P_{FT}	P_{Base}	P_{FT}	P_{Base}	P_{FT}
Qwen2.5-7B (GRPO)	22.1	39.7	53.2	73.8	29.8	33.8	32.5	47.5	6.7	6.7
Qwen2.5-7B (SFT)	22.1	40.1	53.2	69.8	29.8	25.3	32.5	52.5	6.7	10.0
Qwen2.5-1.5B (GRPO)	0.6	18.8	0.6	55.6	3.0	26.8	0.0	30.0	0.0	3.3
Qwen2.5-1.5B (SFT)	0.6	11.0	0.6	36.2	3.0	13.1	0.0	15.0	0.0	0.0
Qwen2.5-Math-7B (GRPO)	19.3	41.0	60.2	79.8	30.3	32.8	32.5	72.5	13.3	16.7
Qwen2.5-Math-7B (SFT)	19.3	43.9	60.2	76.4	30.3	30.8	32.5	50.0	13.3	20.0

Table 7. Detailed performance score of base models (P_{Base}) and fine-tuned models (P_{FT}) across five mathematical benchmarks, served as complementary of Table 1.

the details of the training parameters. We saved 8 LoRA checkpoints during the SFT process. We then evaluated the performance of Temporal Sampling using LoRA checkpoints, comparing its performance against a baseline that sampled only from the final checkpoint. The comparison was based on the Pass@k and Maj@k metrics on the AIME benchmark.

Our findings, illustrated in Figure 10, reveal that Temporal Sampling implemented with LoRA checkpoints outperforms sampling only from the final checkpoint for both Pass@k and Maj@k. This demonstrates that the enhanced sampling performance of Temporal Sampling could be achieved with the considerably reduced storage footprint afforded by LoRA. This makes Temporal Sampling with LoRA a more resource-efficient approach for leveraging checkpoint diversity.

G.4. Comparison of Mixture of Models

In this section, we evaluate our proposed Temporal Sampling against the Mixture of Models, which combines outputs from different foundation models to answer each question collaboratively. To compare sampling efficiencies, we construct a model pool containing three models: our RL-trained final checkpoint (Qwen2.5-7B-Base), Llama 3.1-8B, and DeepSeek-Math-7B-Instruct. We apply Temporal Sampling (with $t = 3$) and the mixture strategy by sampling in a round-robin manner over the pool, then measure the majority voting performance $\mathrm{Maj@}k$. As shown in Figure 11, Temporal Sampling achieves higher sampling performance than the mixture of models under the same computational budget. At $\mathrm{Maj@}64$, Temporal Sampling outperforms the mixture approach by over 4, 9, and 9 points on the AIME24, AMC, and AIME25 benchmarks, respectively.

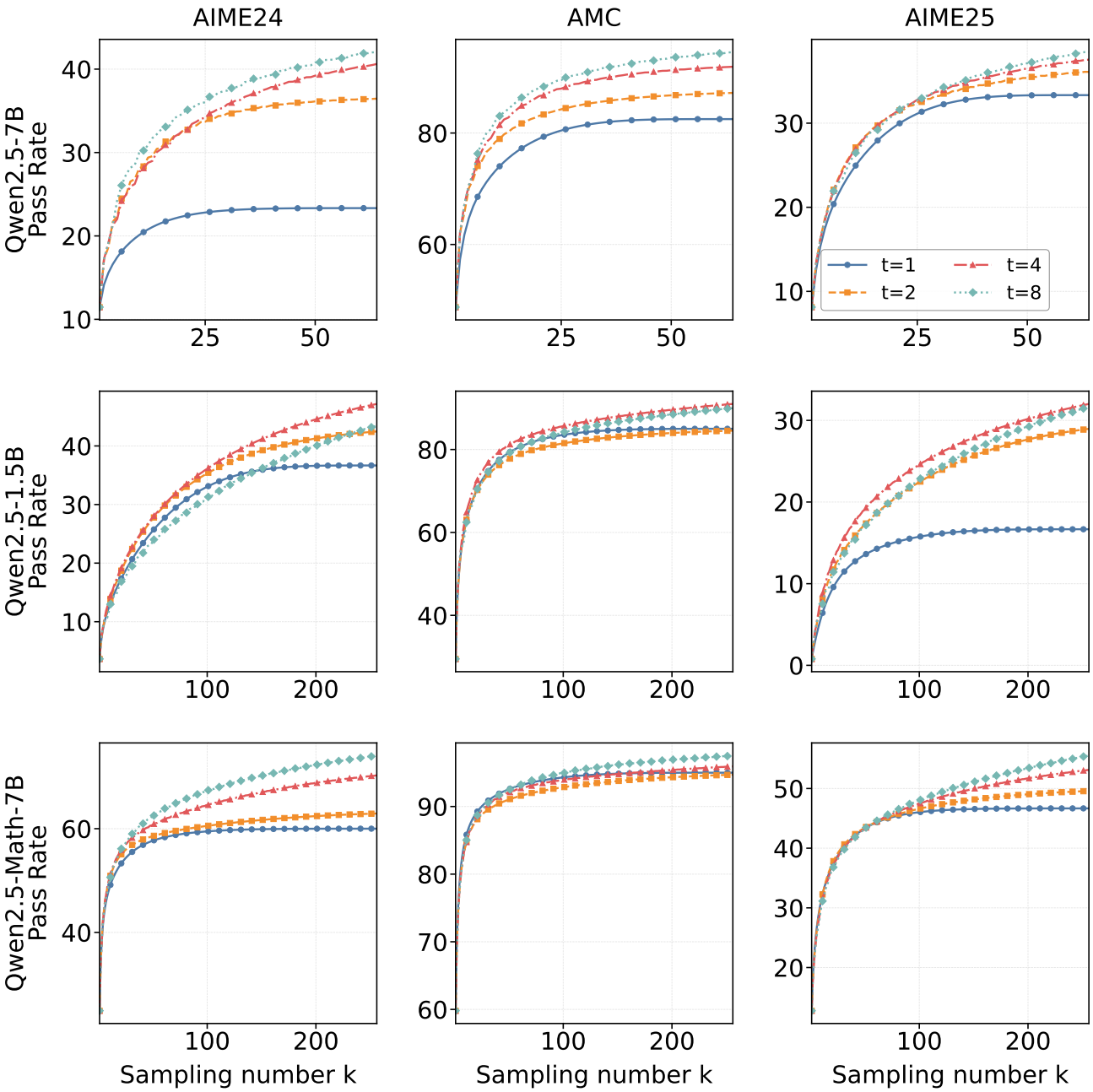

Figure 7. Pass rate of different models with their RL training checkpoints on the AIME2024, AMC, and AIME2025 benchmarks when using Temporal Sampling. The case $t = 1$ represents the baseline of standard $Pass@k$ sampling on the final checkpoint. Our proposed Temporal Sampling outperforms the baseline by more on AIME2024, AMC, and AIME2025.

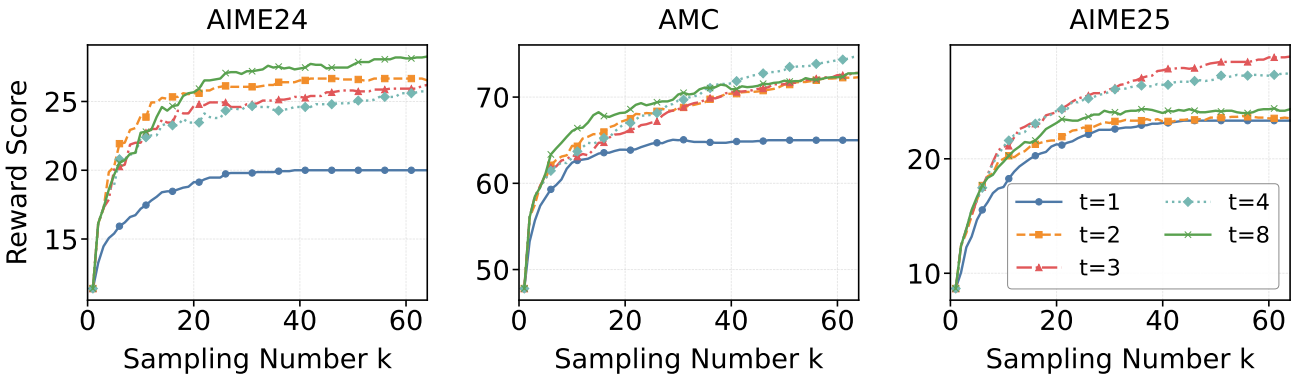

Figure 8. BoN (Best-of-N) decoding on the AIME2024, AMC, and AIME2025 benchmarks using Temporal Sampling. Qwen2.5-Math-PRM-72B is used as the process reward model. We choose the answer with the highest reward as the final answer. The case $t = 1$ represents the baseline of standard BoN on the final checkpoint. Our proposed Temporal Sampling with $t = 8$ checkpoints outperforms the baseline by more than 7, 8, and 1 percentage points on AIME2024, AMC, and AIME2025, respectively, when sampling 64 responses.

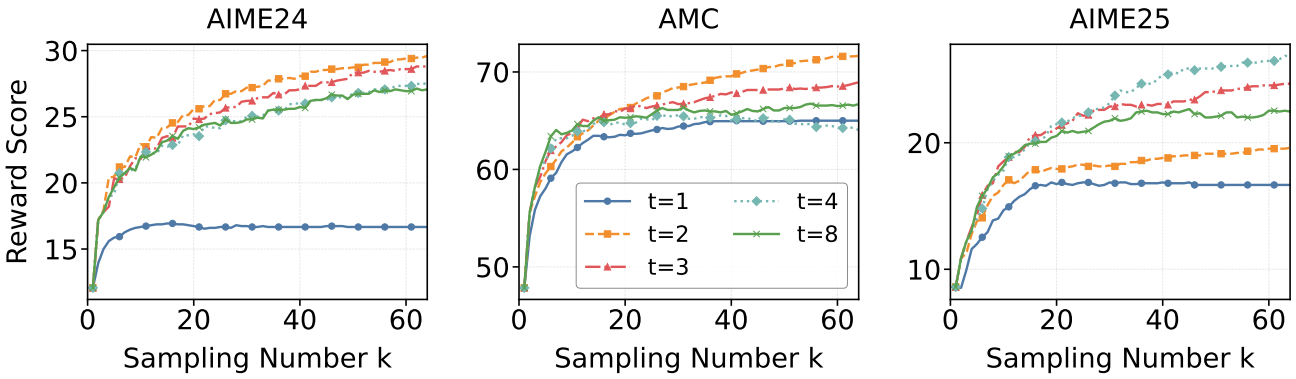

Figure 9. BoN (Best-of-N) decoding on the AIME2024, AMC, and AIME2025 benchmarks using Temporal Sampling. Qwen2.5-Math-PRM-7B is used as the process reward model. We choose the answer with the highest reward as the final answer. The case $t = 1$ represents the baseline of standard BoN on the final checkpoint. Our proposed Temporal Sampling with $t = 8$ checkpoints outperforms the baseline by more than 10, 2, and 5 percentage points on AIME2024, AMC, and AIME2025, respectively, when sampling 64 responses.

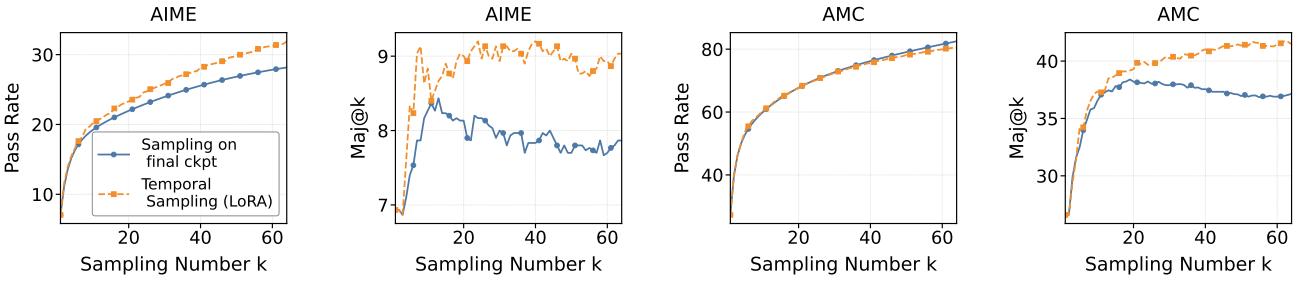

Figure 10. Performance of Temporal Sampling using 8 checkpoints from LoRA SFT of Qwen2.5-7B. Results on the AIME24 and AMC demonstrate that Temporal Sampling with LoRA checkpoints surpasses the baseline (sampling only from the final checkpoint) for both $Pass@k$ and Maj@k.

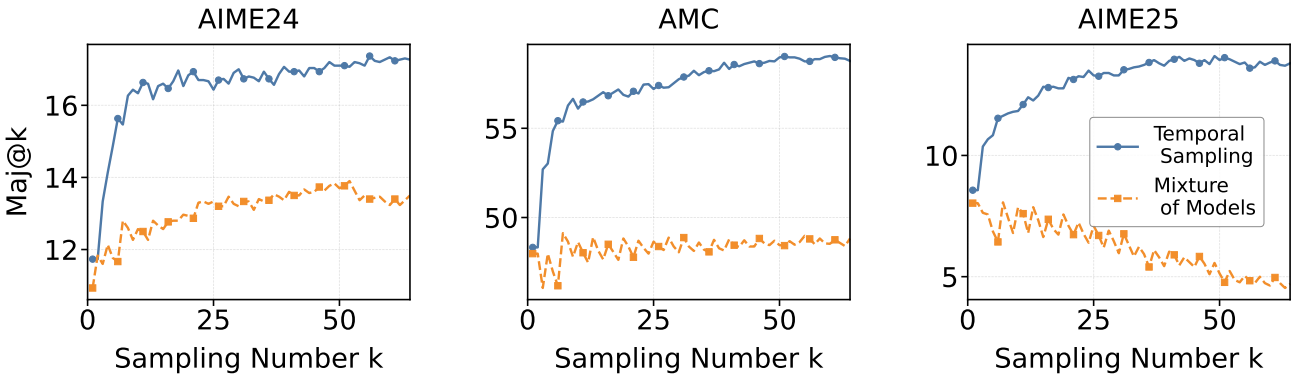

Figure 11. Maj@k comparison between Temporal Sampling ($t = 3$) and a Mixture of Models (MoM) approach on the AIME2024, AMC, and AIME2025 benchmarks. For MoM, the model pool included the Qwen2.5-7B-Base final RL checkpoint, Deepseek-Math-7B-Instruct, and Llama-3.1-8B-Instruct. Temporal Sampling outperforms the MoM approach by more than 4, 9, and 9 percentage points on AIME2024, AMC, and AIME2025, respectively, when sampling 64 responses.