TTVS: Boosting Self-Exploring Reinforcement Learning via Test-time Variational Synthesis

Sikai Bai, Haoxi Li, Jie Zhang†, Yongjiang Liu, Song Guo†
The Hong Kong University of Science and Technology
{sbaiae, hligb, yliunr}@connect.ust.hk,
{csejzhang, songguo}@cse.ust.hk

Abstract

Despite significant advances in Large Reasoning Models (LRMs) driven by reinforcement learning with verifiable rewards (RLVR), this paradigm is fundamentally limited in specialized or novel domains where such supervision is prohibitively expensive or unavailable, posing a key challenge for test-time adaptation. While existing test-time methods offer a potential solution, they are constrained by learning from static query sets, risking overfitting to textual patterns. To address this gap, we introduce Test-Time Variational Synthesis (TTVS), a novel framework that enables LRMs to self-evolve by dynamically augmenting the training stream from unlabeled test queries. TTVS comprises two synergistic modules: (1) Online Variational Synthesis, which transforms static test queries into a dynamic stream of diverse, semantically-equivalent variations, enforcing the model to learn underlying problem logic rather than superficial patterns; (2) Test-time Hybrid Exploration, which balances accuracy-driven exploitation with consistency-driven exploration across synthetic variants. Extensive experiments show TTVS yields superior performance across eight model architectures. Notably, using only unlabeled test-time data, TTVS not only surpasses other test-time adaptation methods but also outperforms state-of-the-art supervised RL-based techniques trained on vast, high-quality labeled data.

1 Introduction

Recent advances in Large Reasoning Models (LRMs) have demonstrated remarkable capabilities, achieving significant breakthroughs on complex reasoning tasks such as mathematics (Yang et al., 2024; Guo et al., 2025) and programming (Hui et al., 2024; Khan et al., 2025). A primary driver behind this progress is the paradigm of Reinforcement Learning with Verifiable Rewards (RLVR), which has been proven by frontier models like OpenAI-o1 (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025) to endow LLMs with advanced reasoning capabilities. A key factor in these developments is the availability of large-scale supervised datasets (Bai et al., 2025a; Dubey et al., 2024), which incorporate instructions and corresponding ground-truth labels for reward computation.

However, in specialized domains such as clinical diagnostics (Ullah et al., 2024; McDuff et al., 2025) and aerospace engineering (Liu et al., 2025a; Connolly, 2025), collecting high-quality data and verifiable supervision remains a significant challenge. Acquiring such human-annotated data is often prohibitively expensive and difficult to scale (Villalobos et al., 2024; Bai et al., 2024b). While a potential alternative is to use external LLMs for cost-effective data generation, this approach usually depends on access to highly capable expert-level models, which are not always readily available (Yang et al., 2023; Goldie et al.). As complex and unlabeled problems continuously emerge, this reliance on external supervision becomes a fundamental bottleneck (Bai et al., 2024a), motivating a transition to what Cheng et al. (2026) call the "era of experience", where models are enabled to self-evolve without direct human oversight.

Building upon this vision, it naturally motivates a promising direction in which LRMs autonomously improve via RL on unlabeled data. In this paper, we focus on a particularly potent mode of such self-evolution: adaptation to test-time data (Shu et al., 2022; Snell et al., 2024; Zhang et al., 2025b). While recent methods have successfully enabled test-time RL by generating rewards through majority voting (Zuo et al., 2025), they are fundamentally constrained by learning from a fixed, predefined set of test queries. Namely, this approach still operates on a static test set, which may lead to overfitting on superficial textual patterns rather than the underlying problem logic. This

†Corresponding authors.

Findings of the Association for Computational Linguistics: ACL 2026, pages 32651–32665

raises a pivotal question: *if reward signal can be generated on-the-fly, can the training data itself be dynamically augmented to enhance performance?*

To answer this question, we propose **Test-Time Variational Synthesis (TTVS)**, a novel framework that boosts self-exploring RL by actively transforming static test queries into a dynamic training stream. Specifically, TTVS is composed of two synergistic modules: (1) The *Online Variational Synthesis* module leverages the model's own consensus to generate semantically-equivalent but differently phrased queries, applying an online filter to ensure quality and diversity. This process enforces the model to learn the underlying problem logic rather than superficial patterns. (2) The *Test-time Hybrid Exploration* module utilizes a dual-mode update strategy, Intra-Group Exploration (IGE) for accuracy-driven exploitation and Cross-Group Exploration (CGE) for consistency-driven exploration, to enable robust and generalizable learning within this augmented data space. Moreover, TTVS is agnostic to policy optimization algorithms and can be flexibly incorporated into other methods, such as GRPO (Shao et al., 2024), OPO (Hao et al., 2025), and DAPO (Yu et al., 2025). Extensive experiments across various benchmarks and model families (Qwen3, Qwen2.5, LLaMA) demonstrate that TTVS achieves superior performance in mathematical reasoning using solely unlabeled test-time data. Remarkably, TTVS not only surpasses other test-time counterparts but also outperforms state-of-the-art RL-based post-training methods, where the latter depend on large-scale, annotated datasets.

2 Related Works

Reinforcement Learning (RL) (Sutton et al., 1998) is a pivotal technique for advancing the reasoning capabilities of Large Language Models (LLMs). Early research focused on Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017; Ouyang et al., 2022), which aligns LLMs with human values by training a reward model based on human preference data. Despite its efficacy, RLHF is often resource-intensive, necessitating extensive human annotation (Gao et al., 2023; Bai et al., 2025b). Recently, Reinforcement Learning with Verifiable Rewards (RLVR) (Schulman et al., 2017; Liu et al., 2025b; Li et al., 2025; Wu et al., 2025) has emerged as a more scalable alternative for domains with objective correctness criteria (e.g., mathematics reasoning) using rule-based reward functions instead of humans. Despite the progress shown by state-of-the-art RL algorithms (e.g., GRPO (Shao et al., 2024), DAPO (Yu et al., 2025)), a fundamental challenge persists: these models are trained via supervision, but their inference during test-time necessitates CoT reasoning for novel and unseen problems.

Test-time Scaling (TTS) (Zhang et al., 2025b) aims to improve model performance by leveraging additional computational resources at inference time. It is primarily divided into two paradigms: *1) Test-Time Inference (TTI)* (Wang et al., 2022; Yuan et al., 2024) enhances output quality by generating multiple solution candidates and selecting the best one via a scoring function or iterative refinement. *1) Test-Time Training (TTT)* (Osowiechi et al., 2024; Zhang et al., 2025c) adapts a pre-trained model to distribution shifts at inference time, commonly employing self-supervised objectives such as entropy minimization and pseudo-labeling. However, prior work has primarily focused on video generation (Dalal et al., 2025) and understanding (Liu et al., 2024). Existing TTS work on the integration of LLMs and reinforcement learning remains limited (Fang et al., 2025), and prevailing methods rely on offline training with fixed test sets, which prevents the model from achieving continual self-improvement that adapts to its current capabilities.

Self-improvement Reinforcement Learning aims to overcome the cost and label requirements of traditional Reinforcement Learning (RL) (Tian et al., 2024). Current approaches largely fall into two categories: *1) Synthetic data generation* (Goldie et al., 2025; Kim et al., 2025) utilizes powerful expert models (e.g., DeepSeek R1 and GPT-4) to generate synthetic data. While effective, this approach incurs substantial computational costs and relies on external models. *2) Self-rewarding metric* (Hao et al., 2026; Zhang et al., 2025a; Lin et al., 2025; Zhang et al., 2026) uses predefined rules to generate preference pairs for RL training or adopts a majority-vote-based self-rewarding/certainty mechanism in an unsupervised RL setting. However, a critical flaw of the self-rewarding strategy is its reliance on the fixed dataset.

3 Preliminaries

3.1 Reinforcement Learning for LLMs

We frame the task of improving LLM reasoning as a reinforcement learning problem. An LLM is treated as a policy π_θ, parameterized by θ,

which generates a sequence of tokens (a solution) $o = (t_1, t_2, \ldots, t_L)$ in response to an input query q. The goal of RL is to optimize the parameters θ to maximize the expected reward for the generated solutions:

$$\max_{\theta} \mathbb{E}_{q\sim\mathcal{D}, o\sim\pi_\theta(\cdot|q)}[R(o, q)], \tag{1}$$

where $\mathcal{D}$ is the data distribution and $R(o, q)$ is a reward function that evaluates the quality of the solution o for the query q.

3.2 Group Relative Policy Optimization

In this work, we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), an efficient RL algorithm well-suited for LLM training as it forgoes an explicit critic model. For a given query q, GRPO samples a group of G outputs $\{o_1, \ldots, o_G\}$ from the current policy π_θ. The policy is then updated by maximizing the following objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\mathcal{L}_i(\theta)\right], \tag{2}$$

where $\mathcal{L}_i(\theta)$ is the per-sample objective, often a clipped surrogate objective similar to PPO, weighted by the advantage A_i:

$$\mathcal{L}_i(\theta) = \min\left(\rho_i(\theta), \text{clip}(\rho_i(\theta), 1-\epsilon, 1+\epsilon)\right)A_i). \tag{3}$$

Here, $\rho_i(\theta) = \pi_\theta(o_i|q)/\pi_{\theta_{\text{old}}}(o_i|q)$ is the probability ratio. A key feature of GRPO is that it estimates the advantage A_i directly from the group's rewards $\{r_1, \ldots, r_G\}$ by treating the mean reward as a baseline, which is both computationally efficient and effective:

$$A_i = \frac{r_i - \text{mean}(\{r_j\}_{j=1}^{G})}{\text{std}(\{r_j\}_{j=1}^{G}) + \delta}, \tag{4}$$

where δ is a small constant for numerical stability.

3.3 Reward Signal via Majority Voting

In the test-time setting, ground-truth labels are unavailable, making reward computation a central challenge. Following TTRL (Zuo et al., 2025), we generate a reward signal by estimating a pseudo-label via majority voting. Given a query q, we first sample N candidate solutions $\{o_1, \ldots, o_N\}$ from the policy. We then extract a final answer $a_i = \text{ExtractAnswer}(o_i)$ from each solution. The most frequently occurring answer ($\text{MajorityVote}(\cdot)$) is designated as the pseudo-label y^*:

$$y^* = \arg\max_{a} \sum_{i=1}^{N} \mathbb{I}(a_i = a), \tag{5}$$

where $\mathbb{I}(\cdot)$ is the indicator function. The reward r_i for each solution o_i is then determined by its agreement with this pseudo-label:

$$r_i = R(o_i, y^*) = \begin{cases} 1, & \text{if } a_i = y^* \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

This mechanism provides a robust, self-generated reward signal that enables RL training in the absence of explicit supervision.

4 Methodology

In this work, we propose TTVS, a novel test-time RL framework that actively augments the training data and employs a dual-update mechanism to balance accuracy and generalization. Our method enables the model not only to adapt to the test data but also to learn the invariant underlying logic of the problems. The overall pipeline is depicted in Figure 1 and consists of two primary stages: (1) Online Variational Synthesis and (2) Test-time Hybrid Exploration.

4.1 Online Variational Synthesis

The first stage of our framework focuses on generating a rich and diversified set of training instances from the original, static test query. This is achieved through a three-step process: pseudo-label generation, variational data augmentation, and online filtering.

We first generate a pseudo-label for each original query q from the test set. Given a query q, we sample N candidate solutions (rollouts) $\{o_1, o_2, \ldots, o_N\}$ from the current policy π_θ. An answer a_i is extracted from each rollout, and a consensus answer y_q^* is determined via majority voting. This consensus answer serves as the pseudo-label for the query q.

$$\{o_1, \ldots, o_N\} \sim \pi_\theta(\cdot|q), \tag{7}$$

$$y_q^* = \text{MajorityVote}((a_i)_{i=1}^{N}), \tag{8}$$

where pseudo-label y_q^* is the cornerstone for both reward calculation and the subsequent data synthesis process.

At the heart of our method lies the concept of online variational synthesis. Instead of solely relying on the original query q, we prompt the policy model π_θ to generate k new queries that are semantically consistent with q and share the same answer y_q^*, but differ in their surface-level expression. We design a specific prompt $\mathcal{P}$ that instructs the model

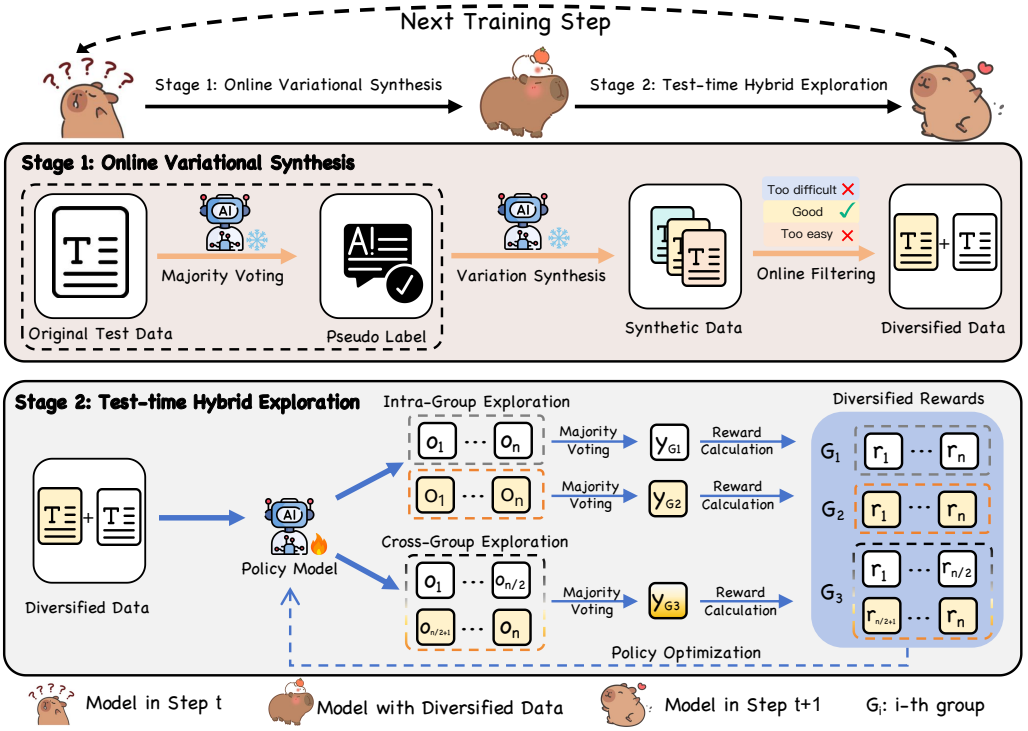


Figure 1: The schematic illustration of the Test-time Variational Synthesis (TTVS). In **Stage 1: Online Variational Synthesis.** Pseudo labels are generated from test data to synthesize diversified samples, which are then online filtered for quality and diversity. In **Stage 2: Test-time Hybrid Exploration.** A policy model is optimized on this augmented data through a hybrid intra- and cross-group exploration strategy, culminating in diversified rewards that guide the model's optimization.

to paraphrase the original problem. The generation process is as follows:

$$\{q'_1, q'_2, \ldots, q'_k\} \sim \pi_\theta(\cdot|\mathcal{P}, q,\ y^*_q). \tag{9}$$

This synthesis process transforms a single data point into a cluster of semantically equivalent problems $\{q, q'_1, \ldots, q'_k\}$. Training on this augmented data encourages the model to develop a more robust and abstract understanding of the reasoning task, moving beyond mere pattern matching of the input text.

To ensure the quality of the augmented data, we introduce a crucial online filtering stage. Not all synthesized queries are retained. A query cluster originating from q is only generated and used for training if it satisfies two conditions. First, the initial group accuracy for the original query q, denoted $\text{acc}(q)$, must lie within a predefined difficulty range $[\tau_{\text{low}}, \tau_{\text{high}}]$:

$$\text{acc}(q) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(a_i = y^*_q) \in [\tau_{\text{low}}, \tau_{\text{high}}]. \tag{10}$$

This condition ensures that the model focuses on problems that are neither trivial nor intractable. Second, each synthesized query q'_j must not exceed a maximum token length $L_{\max}$:

$$\texttt{Length}(q'_j) \leq L_{\max}, \tag{11}$$

only the data satisfying these criteria proceed to the policy update stage, thus curating a training batch of suitable difficulty and quality. We provide more details for online filtering in Appendix A.1

4.2 Test-time Hybrid Exploration

Having generated a high-quality batch of original and variational queries $\mathcal{D}_{\text{batch}}$, the second stage performs policy updates using our proposed hybrid exploration mechanism. Two complementary update modes, Intra-Group Exploration and Cross-Group Exploration, are designed to execute independently within each training step. The rationale behind this dual-mode approach is to synergistically improve both the model's accuracy on specific problem instances and its consistency across semantic variations.

The first mode, Intra-Group Exploration (**IGE**), focuses on maximizing performance on each individual problem by operating within the confines of a single query's generated outputs. For every

query $\tilde{q} \in \mathcal{D}_{\text{batch}}$ (where $\tilde{q}$ can be an original query q or a variational query q'_j), we perform a full, independent GRPO update. The process begins by performing a complete rollout to generate N solutions, from which a pseudo-label $y^*_{\tilde{q}}$ is obtained via intra-group majority voting. Subsequently, the advantage for each rollout is calculated based on this intra-group reward signal $R(o_i, y^*_{\tilde{q}})$, and the policy parameters θ are updated by optimizing the GRPO objective for that specific query $\tilde{q}$.

This mode represents an accuracy-driven exploitation of the data, as it treats each query as a distinct optimization target to be mastered. It pushes the model to find the correct solution for that specific phrasing. The total update from this mode can be seen as the sum of independent policy gradients for all queries in the batch.

The second mode, Cross-Group Exploration (**CGE**), is designed to explicitly enforce consistency by operating across the semantically-equivalent problem cluster $\{q, q'_1, \ldots, q'_k\}$. Instead of performing separate rollouts for each query, we create a mixed pool of rollouts by sampling a fraction of solutions from each query in the cluster:

$$\mathcal{O}_{\text{mix}} = \bigcup_{j=0}^{k} \left\{ \text{Sample}_{i=1}^{N/(k+1)} \left(o_i \sim \pi_\theta(\cdot|q'_j) \right) \right\}, \tag{12}$$

where q'_0 denotes the original query q. Critically, a single **cross-group majority vote** is performed over this entire mixed set $\mathcal{O}_{\text{mix}}$ to derive a joint pseudo-label y^*_{mix}. This joint label represents the model's most robust consensus across all variational expressions of the problem. The advantage for every rollout $o_i \in \mathcal{O}_{\text{mix}}$ is then calculated with respect to this single, consistency-enforcing label y^*_{mix}. The subsequent GRPO update, therefore, optimizes the policy based on a unified, cross-group reward signal, compelling it to become self-consistent across the entire semantic cluster. This mode provides a form of consistency-driven exploration, as it regularizes the policy and ensures its reasoning process is invariant to superficial changes in the problem statement.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate model performance on a wide range of mathematical reasoning benchmarks. These include MATH-500 (Hendrycks et al., 2021), a curated collection of 500 competition-level problems from the MATH dataset; AIME-2024 (Li et al., 2024), comprising challenging problems from the 2024 American Invitational Mathematics Examination; AMC-2023 (Li et al., 2024), a dataset sourced from the American Mathematics Competitions that contains 83 samples comprising a series of progressively challenging mathematical tests designed for middle and high school students; and GPQA (Rein et al., 2024), a high-quality and exceptionally challenging subset of the Graduate-Level Google-Proof Question Answering benchmark that curated from domains such as physics, chemistry, and biology.

Models. To assess the generalizability of our proposed method, we conducted extensive experiments across three distinct model families, encompassing a wide range of parameter scales. Specifically, the models we experiment with are as follows. *1) Qwen3 Family* (Yang et al., 2025): Qwen3-1.7B-Instruct, Qwen3-4B, and Qwen3-8B. *2)Qwen2.5 Family* (Yang et al., 2024): Qwen2.5-Instruct-1.5B, Qwen2.5-Instruct-3B, and Qwen2.5-Instruct-7B. *3) LLaMA Family* (Dubey et al., 2024): LLaMA-3.2-3B-Instruct and LLaMA3.1-8B.

Baselines. We compare our method with the following methods: 1) RL Post-Trained Models: they are leading models with architectures similar to our own, which have undergone extensive reinforcement learning on large-scale labeled data, including DeepSeek-R1-Distill (1.5B & 7B) (Guo et al., 2025), SimpleRL-Zero-7B (Zeng et al., 2025), and OpenReasoner-Zero-7B (Hu et al., 2025). It is worth noting that our TTVS operates under a test-time adaptation paradigm using only unlabeled data, which represents a fundamental difference from these methods. 2) TTRL (Zuo et al., 2025): similar to our TTVS, this method also utilizes test-time reinforcement learning; however, it only estimates rewards through majority voting on original unlabeled data during training.

Implementation Details. For the implementation of TTVS, we employ GRPO as the optimization strategy across all benchmarks. The policy model is optimized using the AdamW optimizer with a cosine learning rate schedule, peaking at 5×10^{-7}. During the rollout phase, 32 responses are sampled (temperature = 0.6) for the voting-based label estimation. By default, we set L_{max} is 1024, τ_{low} is 0.125 and τ_{high} equals 0.875. To ensure a fair comparison with the TTRL, the maximum generation length is constrained to 3072 tokens. For evaluation metrics, we evaluate model

Methods	MATH500	AIME2024	AMC2023	GPQA	Average
RL Post-trained Models w / Labeled Data					
DeepSeek-R1-Distill-1.5B (800K)	52.2	2.5	21.7	14.6	22.8
DeepSeek-R1-Distill-7B (800K)	60.1	10.0	26.2	25.7	30.5
OpenReasoner-Zero-7B (129K)	79.2	13.3	47.0	28.4	42.0
SimpleRL-Zero-7B (8.9K)	78.2	26.7	60.2	27.6	48.2
Qwen3 Family					
Qwen3-1.7B-Instruct	59.4	3.3	28.9	13.1	26.2
w / TTRL	73.4$_{(+14.0)}$	15.2$_{(+11.9)}$	54.2$_{(+25.3)}$	22.2$_{(+9.1)}$	41.3$_{(+15.1)}$
w / TTVS (Ours)	**80.2**$_{(+20.8)}$	**23.3**$_{(+20.0)}$	**61.4**$_{(+32.5)}$	**26.3** $_{(+16.7)}$	**47.5**$_{(+21.3)}$
Qwen3-4B	64.0	10.0	26.5	26.3	31.7
w / TTRL	85.0$_{(+21.0)}$	26.7$_{(+16.7)}$	61.4$_{(+34.9)}$	43.0$_{(+16.7)}$	54.0$_{(+22.3)}$
w / TTVS (Ours)	**90.3**$_{(+26.3)}$	**36.7**$_{(+26.7)}$	**71.1**$_{(+44.6)}$	**48.9** $_{(+21.7)}$	**61.5**$_{(+29.8)}$
Qwen3-8B	82.2	26.9	57.8	48.0	53.7
w / TTRL	89.2$_{(+7.0)}$	46.7$_{(+19.8)}$	68.6$_{(+10.8)}$	53.0$_{(+5.0)}$	65.4$_{(+10.7)}$
w / TTVS (Ours)	**92.6**$_{(+10.4)}$	**50.0**$_{(+23.1)}$	**72.3**$_{(+14.5)}$	**56.1** $_{(+8.1)}$	**67.8**$_{(+14.1)}$
Qwen2.5 Family					
Qwen2.5-Instruct-1.5B	54.4	0.0	24.1	16.7	23.8
w / TTRL	62.2$_{(+7.8)}$	3.3$_{(+3.3)}$	32.5$_{(+8.4)}$	25.3$_{(+8.6)}$	30.8$_{(+7.1)}$
w / TTVS (Ours)	**65.8**$_{(+11.4)}$	**3.3**$_{(+3.3)}$	**36.1**$_{(+12.0)}$	**28.3** $_{(+11.6)}$	**33.4**$_{(+9.6)}$
Qwen2.5-Instruct-3B	64.2	3.3	32.5	29.3	32.3
w / TTRL	71.2$_{(+7.0)}$	6.7$_{(+3.4)}$	39.8$_{(+7.3)}$	33.3$_{(+4.0)}$	37.8$_{(+5.4)}$
w / TTVS (Ours)	**75.0**$_{(+10.8)}$	**6.7**$_{(+3.4)}$	**42.2**$_{(+9.7)}$	**36.4** $_{(+7.1)}$	**40.1**$_{(+7.8)}$
Qwen2.5-Instruct-7B	70.4	10.2	41.0	36.4	39.5
w / TTRL	77.6$_{(+7.2)}$	13.3$_{(+3.1)}$	44.6$_{(+3.6)}$	48.5$_{(+12.1)}$	46.0$_{(+6.5)}$
w / TTVS (Ours)	**79.4**$_{(+9.0)}$	**16.7**$_{(+6.5)}$	**47.2**$_{(+6.2)}$	**51.5** $_{(+15.1)}$	**48.7**$_{(+9.2)}$
LLaMA Family					
LLaMA-3.2-3B-Instruct	40.8	0.8	15.1	12.5	17.3
w / TTRL	51.0$_{(+10.2)}$	3.3$_{(+2.5)}$	25.3$_{(+10.2)}$	16.1$_{(+3.6)}$	23.9$_{(+6.6)}$
w / TTVS (Ours)	**54.1**$_{(+13.3)}$	**6.9**$_{(+6.1)}$	**27.9**$_{(+12.8)}$	**17.7** $_{(+5.2)}$	**26.7**$_{(+9.4)}$
LLaMA3.1-8B	48.6	4.6	23.3	30.8	26.8
w / TTRL	63.7$_{(+15.1)}$	10.0$_{(+5.4)}$	32.3$_{(+9.0)}$	34.1$_{(+3.3)}$	35.0$_{(+8.2)}$
w / TTVS (Ours)	**64.9**$_{(+16.3)}$	**10.0**$_{(+5.4)}$	**33.5**$_{(+10.2)}$	**37.3** $_{(+6.5)}$	**36.4**$_{(+9.6)}$

Table 1: Performance comparison of TTVS with state-of-the-art methods on various model architectures. For RL Post-trained Models *w /* Labeled Data $(\cdot)$, $(\cdot)$ indicates the amount of labeled data used during post-training.

performance using pass@1 Score. For each problem, we generate 16 candidate responses using temperature sampling (temperature = 0.6, top-p = 0.95) and assess the correctness of the top-ranked solution. Moreover, we set the warmup steps as 40 for IGE and 60 for CGE on aross four reasoning benchmarks. The number of training episodes was configured to 10, 10, 20, and 40 for MATH500, GPQA, AMC2023 and AIME2024, scaled according to their size. All experiments were conducted using 8 NVIDIA GeForce H800 80GB GPUs.

5.2 Main Results

To illustrate the efficacy of our proposed method, we report the performance comparison of TTVS against several state-of-the-art (SOTA) methods. Our evaluation spans 8 models across 3 distinct model families, with detailed results presented in Table 1. **1) Comparison with RL Post-trained Models with Labeled Data**, TTVS outperforms these methods on most settings. When applied to Qwen2.5-Instruct-7B, TTVS exhibits a clear performance advantage over leading 7B models DeepSeek-R1-Distill-7B, OpenReasoner-Zero-7B, particularly on MATH500 and GPQA benchmark. Furthermore, the efficiency of TTVS is underscored by the results that even our smaller Qwen2.5-Instruct-1.5B model outperforms the larger, post-trained DeepSeek-R1-Distill-7B (e.g., 33.4% vs. 30.5% average score). This indicates that TTVS can unlock latent reasoning abilities more effectively using test-time self-improvement. **2) Compared to the SOTA test-time RL method (TTRL)**, our TTVS demonstrates improved perfor-

Methods	MATH500	AMC2023	GPQA
Qwen3-4B	64.0	26.5	26.3
↪**CGE**	86.4	62.6	43.9
↪**IGE**	88.4	69.8	46.6
↪TTVS	90.3	71.1	48.0

Table 2: Performance analysis of different components in TTVS.

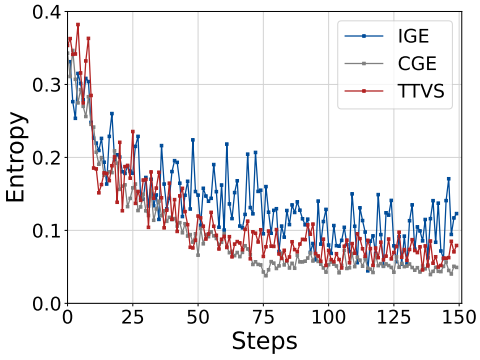


Figure 2: Entropy Curve of TTVS components on AMC2023 using Qwen3-4B.

mance across various benchmarks and model types. Notably, on Qwen3-4B, TTVS yields absolute gains of 5.3% (MATH500), 10.0% (AIME2024), 9.7% (AMC2023), and 5.9% (GPQA). These results demonstrate the effectiveness of TTVS and highlight the potential of self-improvements to facilitate greater exploration using online variational synthesis.

5.3 Ablation Studies

5.3.1 Effectiveness of Components

To validate the contributions of the individual components within our TTVS, we conduct ablation studies on Qwen3-4B. As shown in Table 2, we systematically evaluate the performance of several variants across the MATH500, AMC2023, and GPQA benchmarks. The naive Qwen3-4B is established as the baseline, and we directly perform the inference on various reasoning tasks and have poor performance. We then evaluate two key components of our framework in isolation. First, the Consistency-Guided Exploitation (CGE) module yields substantial performance gains over the baseline (e.g., 86.4% v.s. 64.0% on MATH500). Second, the Intra-group Exploitation (IGE) module achieves even greater improvements, with scores of 88.4%, 69.8%, and 46.6%. It obtains further performance improvements and exhibits persistently high entropy throughout training in Figure 2. Finally, the full TTVS framework integrates both

Methods	MATH500	AIME2024	AMC2023
Qwen3-4B	64.0	10.0	26.5
↪TTVS (*w/* OPO)	89.6	33.3	69.8
↪TTVS (*w/* DAPO)	88.8	36.7	71.6
↪TTVS (*w/* GRPO)	90.3	36.7	71.1

Table 3: Performance comparison of different RL algorithms for TTVS using Qwen3-4B.

CGE and IGE, leveraging their complementary strengths to balance generalization and accuracy. This synergistic combination achieves more steady exploration and the optimal performance across all benchmarks, reaching 90.3% on MATH500, 71.1% on AMC2023, and 48.0% on GPQA, thereby confirming the efficacy of our synergistic exploration.

5.3.2 Impacts of Different RL Algorithms

To assess the versatility of our TTVS, we evaluated its performance when integrated with three distinct reinforcement learning algorithms: GRPO (Shao et al., 2024), OPO (Hao et al., 2025), and DAPO (Yu et al., 2025). GRPO normalizes rewards within a group of sampled responses to calculate a relative advantage. OPO employs a length-weighted average as an optimal reward baseline to stabilize training. DAPO incorporates a suite of techniques, such as clip-higher and a token-mean objective, to enhance stability in large-scale training. As shown in Table 3, TTVS consistently delivers substantial performance improvements over the Qwen3-4B (baseline), regardless of the specific RL optimizer used. Furthermore, the performance of the three RL algorithms was closely aligned, with only minor differences observed. GRPO achieved the highest score on MATH500 (90.3%), while DAPO and GRPO performed identically on AIME2024 (36.7%). This consistent high performance validates the robustness of TTVS and confirms its compatibility with a range of policy optimization strategies.

5.3.3 Computational Cost Analysis

Finally, we conducted a detailed quantitative analysis of the resources used during both test-time training and the inference phase. **1) During Test-Time Training**, as shown in Figure 4, the GPU memory consumption of TTVS is nearly identical to the TTRL baseline phase with the same number of rollouts (32). Even when compared to a TTRL baseline with double the rollouts (64), our TTVS requires substantially less GPU memory while still

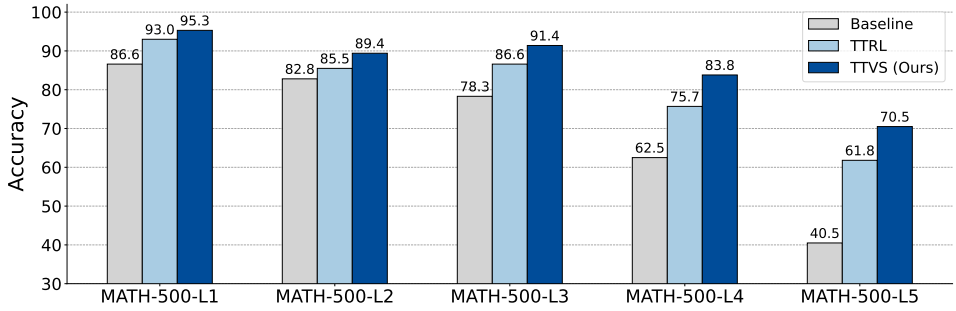


Figure 3: Comparative performance analysis across the five difficulty levels of MATH-500.

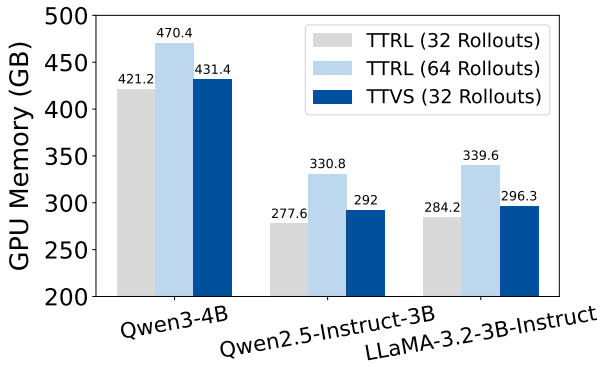


Figure 4: Computational cost during test-time training on Qwen3-4B

Methods	GPU Memory (GB)	Token Length	Throughput (Token/s)
TTRL	401.4	1925.5	15.60
TTVS	398.0	1864.9	14.28

Table 4: Computational cost at inference phase based on Qwen3-4B using 32 rollouts

achieving superior accuracy. This confirms that our method achieves its significant performance improvements without incurring substantial additional computational overhead during test-time training. **2) At the inference phase**, the model produced by TTVS is also more efficient. As shown in Table 4, TTVS generates more concise answers (shorter token length), leading to a slightly lower memory footprint. Overall, these results demonstrate that TTVS offers a superior trade-off between performance and computational cost. Its advantages are derived from the quality of its dynamic data synthesis, not from a greater quantity of computation. We provide more clarification on TTVS's computational trade-off in Appendix A.4.

5.3.4 Impacts of Question Difficulty Levels

To validate the impacts of question difficulty levels for TTVS, we conducted a fine-grained analysis using the MATH-500 dataset. The dataset was partitioned into five subsets corresponding to its annotated difficulty levels (L1 to L5). We then evaluated the performance of the baseline (Qwen3-4B), TTRL, and our TTVS framework on each subset. As illustrated in Figure 3, it shows crucial trend: the performance gap between TTVS and TTRL widens as the complexity of the problems increases. On Level 1 problems, TTVS holds a slight edge over TTRL (95.3% vs. 93.0%). This advantage becomes far more substantial on Level 5 problems, where TTVS achieves an accuracy of 69.5%—with 8.7% and 30.0% performance improvements over TTRL the baseline, respectively. These results demonstrate the superior robustness of TTVS, suggesting that its variational synthesis mechanism effectively compensates for these knowledge gaps, enabling the model to learn and reason more effectively on challenging tasks. We provide more experimental results in Appendix.

6 Conclusion

In this paper, we introduced Test-Time Variational Synthesis (TTVS), a novel framework designed to enhance the self-exploration capabilities of Large Reasoning Models (LRMs) through dynamic data augmentation. TTVS facilitates the online generation of semantically consistent synthetic variations of test queries, thereby enriching the data without the need for additional human annotations. This augmented dataset then enables a hybrid exploration strategy for optimizing the policy model during test-time. Comprehensive experiments demonstrate that TTVS achieves superior performance compared to existing test-time adaptation methods and state-of-the-art RL post-training strategies across a diverse range of mathematical reasoning benchmarks and model architectures, establishing a new performance benchmark for test-time reinforcement learning.

Limitations

While the proposed TTVS method demonstrates strong performance across a range of reasoning benchmarks and model architectures, the present study is subject to several limitations that open avenues for future investigation. First, due to computational constraints, our primary experiments cannot be conducted on some larger-scale LLMs, such as Qwen2.5-32B-Instruct, Qwen3-32B. Validating the scalability and effectiveness of TTVS on such models is an important direction for future work. Second, the scope of this study is confined to language-only models. Consequently, the efficacy of our approach in multimodal architectures remains unexplored. Investigating whether our method can be adapted to achieve competitive performance on test-time vision-language tasks constitutes a promising area for further research.

Acknowledgement

This research was supported by fundings from the Hong Kong RGC General Research Fund (152228/23E, 162161/24E, 162116/25E, 162180/25E), National Natural Science Foundation of China (NSFC) Key Program (No.62532005), Collaborative Research Fund (No. C1042-23GF, No. C5097-25G), NSFC/RGC Collaborative Research Scheme (Grant No. 62461160332 & CRS_HKUST602/24), Research Impact Fund (No. R5011-23F), Areas of Excellence Scheme (AoE/E-601/22-R), and the InnoHK (HKGAI).

References

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, and 1 others. 2025a. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Sikai Bai, Haoxi Li, Jie Zhang, Zicong Hong, and Song Guo. 2025b. Diep: Adaptive mixture-of-experts compression through differentiable expert pruning. *arXiv preprint arXiv:2509.16105*.

Sikai Bai, Shuaicheng Li, Weiming Zhuang, Jie Zhang, Kunlin Yang, Jun Hou, Shuai Yi, Shuai Zhang, and Junyu Gao. 2024a. Combating data imbalances in federated semi-supervised learning with dual regulators. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 10989–10997.

Sikai Bai, Jie Zhang, Song Guo, Shuaicheng Li, Jingcai Guo, Jun Hou, Tao Han, and Xiaocheng Lu. 2024b. Diprompt: Disentangled prompt tuning for multiple latent domain generalization in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27284–27293.

Zihao Cheng, Zeming Liu, Yingyu Shan, Xinyi Wang, Xiangrong Zhu, Yunpu Ma, Hongru Wang, Yuhang Guo, Wei Lin, and Yunhong Wang. 2026. Mem^{2}evolve: Towards self-evolving agents via co-evolutionary capability expansion and experience distillation. *arXiv preprint arXiv:2604.10923*.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Brian J Connolly. 2025. Development of an aerospace engineering evaluation set for large language model benchmarking. In *AIAA SCITECH 2025 Forum*, page 0702.

Karan Dalal, Daniel Koceja, Jiarui Xu, Yue Zhao, Shihao Han, Ka Chun Cheung, Jan Kautz, Yejin Choi, Yu Sun, and Xiaolong Wang. 2025. One-minute video generation with test-time training. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 17702–17711.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Wenkai Fang, Shunyu Liu, Yang Zhou, Kongcheng Zhang, Tongya Zheng, Kaixuan Chen, Mingli Song, and Dacheng Tao. 2025. Serl: Self-play reinforcement learning for large language models with limited data. *arXiv preprint arXiv:2505.20347*.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR.

Anna Goldie, Azalia Mirhoseini, Hao Zhou, Irene Cai, and Christopher D Manning. Synthetic data generation & multi-step rl for reasoning & tool use, 2025. *URL https://arxiv. org/abs/2504.04736*.

Anna Goldie, Azalia Mirhoseini, Hao Zhou, Irene Cai, and Christopher D Manning. 2025. Synthetic data generation & multi-step rl for reasoning & tool use. *arXiv preprint arXiv:2504.04736*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Yaru Hao, Li Dong, Xun Wu, Shaohan Huang, Zewen Chi, and Furu Wei. 2025. On-policy rl with optimal reward baseline. *arXiv preprint arXiv:2505.23585*.

Zhezheng Hao, Hong Wang, Jian Luo, Jianqing Zhang, Yuyan Zhou, Qiang Lin, Can Wang, Hande Dong, and Jiawei Chen. 2026. Recreate: Reasoning and creating domain agents driven by experience. *arXiv preprint arXiv:2601.11100*.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, and 1 others. 2025. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*.

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

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. 2025. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, and 1 others. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Ruhma Khan, Sumit Gulwani, Vu Le, Arjun Radhakrishna, Ashish Tiwari, and Gust Verbruggen. 2025. Llm-guided compositional program synthesis. *arXiv preprint arXiv:2503.15540*.

Jungwoo Kim, Minsang Kim, and Sungjin Lee. 2025. Sedi-instruct: Enhancing alignment of language models through self-directed instruction generation. *arXiv preprint arXiv:2502.04774*.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, and 1 others. 2024. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9.

Mengdi Li, Jiaye Lin, Xufeng Zhao, Wenhao Lu, Peilin Zhao, Stefan Wermter, and Di Wang. 2025. Curriculum-rlaif: Curriculum alignment with reinforcement learning from ai feedback. *arXiv preprint arXiv:2505.20075*.

Jiaye Lin, Yifu Guo, Yuzhen Han, Sen Hu, Ziyi Ni, Licheng Wang, Mingguang Chen, Hongzhang Liu, Ronghao Chen, Yangfan He, and 1 others. 2025. Seagent: Self-evolution trajectory optimization in multi-step reasoning with llm-based agents. *arXiv preprint arXiv:2508.02085*.

Beiming Liu, Zhizhuo Cui, Siteng Hu, Xiaohua Li, Haifeng Lin, and Zhengxin Zhang. 2025a. Llm evaluation based on aerospace manufacturing expertise: Automated generation and multi-model question answering. *arXiv preprint arXiv:2501.17183*.

Weihuang Liu, Xi Shen, Haolun Li, Xiuli Bi, Bo Liu, Chi-Man Pun, and Xiaodong Cun. 2024. Depth-aware test-time training for zero-shot video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19218–19227.

Yule Liu, Heyi Zhang, Jinyi Zheng, Zhen Sun, Zifan Peng, Tianshuo Cong, Yilong Yang, Xinlei He, and Zhuo Ma. 2025b. Grpo privacy is at risk: A membership inference attack against reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2511.14045*.

Daniel McDuff, Mike Schaekermann, Tao Tu, Anil Palepu, Amy Wang, Jake Garrison, Karan Singhal, Yash Sharma, Shekoofeh Azizi, Kavita Kulkarni, and 1 others. 2025. Towards accurate differential diagnosis with large language models. *Nature*, pages 1–7.

David Osowiechi, Gustavo A Vargas Hakim, Mehrdad Noori, Milad Cheraghalikhani, Ali Bahri, Moslem Yazdanpanah, Ismail Ben Ayed, and Christian Desrosiers. 2024. Nc-ttt: A noise constrastive approach for test-time training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6078–6086.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pages 248–260. PMLR.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. 2022. Test-time prompt tuning for zero-shot generalization in vision-language models. *Advances in Neural Information Processing Systems*, 35:14274–14289.

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

Richard S Sutton, Andrew G Barto, and 1 others. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Ye Tian, Baolin Peng, Linfeng Song, Lifeng Jin, Dian Yu, Lei Han, Haitao Mi, and Dong Yu. 2024. Toward self-improvement of llms via imagination, searching, and criticizing. *Advances in Neural Information Processing Systems*, 37:52723–52748.

Ehsan Ullah, Anil Parwani, Mirza Mansoor Baig, and Rajendra Singh. 2024. Challenges and barriers of using large language models (llm) such as chatgpt for diagnostic medicine with a focus on digital pathology–a recent scoping review. *Diagnostic pathology*, 19(1):43.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2024. Position: Will we run out of data? limits of llm scaling based on human-generated data. In *Forty-first International Conference on Machine Learning*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Zhenhe Wu, Jian Yang, Jiaheng Liu, Xianjie Wu, Changzai Pan, Jie Zhang, Yu Zhao, Shuangyong Song, Yongxiang Li, and Zhoujun Li. 2025. Table-r1: Region-based reinforcement learning for table understanding. *arXiv preprint arXiv:2505.12415*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, and 1 others. 2024. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Rui Yang, Lin Song, Yanwei Li, Sijie Zhao, Yixiao Ge, Xiu Li, and Ying Shan. 2023. Gpt4tools: Teaching large language model to use tools via self-instruction. *Advances in Neural Information Processing Systems*, 36:71995–72007.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. 2024. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. 2025. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*.

Qingyang Zhang, Haitao Wu, Changqing Zhang, Peilin Zhao, and Yatao Bian. 2025a. Right question is already half the answer: Fully unsupervised llm reasoning incentivization. *arXiv preprint arXiv:2504.05812*.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenyue Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, and 1 others. 2025b. A survey on test-time scaling in large language models: What, how, where, and how well? *arXiv preprint arXiv:2503.24235*.

Tianyuan Zhang, Sai Bi, Yicong Hong, Kai Zhang, Fujun Luan, Songlin Yang, Kalyan Sunkavalli, William T Freeman, and Hao Tan. 2025c. Test-time training done right. *arXiv preprint arXiv:2505.23884*.

Wenyuan Zhang, Xinghua Zhang, Haiyang Yu, Shuaiyi Nie, Bingli Wu, Juwei Yue, Tingwen Liu, and Yongbin Li. 2026. Expseek: Self-triggered experience seeking for web agents.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, and 1 others. 2025. Ttrl: Test-time reinforcement learning. In *The Thirty-Ninth Annual Conference on Neural Information Processing Systems*.

Filtering thresholds		Qwen3-4B	Qwen2.5-Instruct-3B
$\tau_{low} = 0.0$	$\tau_{high} = 0.875$	86.4	72.0
$\tau_{low} = 0.125$	$\tau_{high} = 0.875$	**90.3**	**75.0**
$\tau_{low} = 0.25$	$\tau_{high} = 0.875$	86.2	72.4
$\tau_{low} = 0.125$	$\tau_{high} = 0.625$	88.4	73.8
$\tau_{low} = 0.125$	$\tau_{high} = 1.0$	88.7	73.2

Table 5: Ablation study for filtering thresholds

Methods	Qwen3-4B	Qwen2.5-Instruct-3B
512	86.1	72.2
1024	90.3	75.0
2048	90.7	74.9

Table 6: Ablation study for Maximum Token Length.

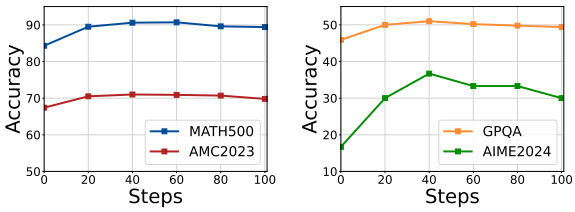
A Appendix

A.1 More Details of the Online Filtering Mechanism

This mechanism is a crucial two-step process designed to curate a training batch of suitable difficulty and quality without verifiable labels. **Step 1: Difficulty Scaffolding (on Original Query q)**: To assess difficulty, we first generate a pseudo-label $y*$ for the original query q via majority voting. We then calculate the initial group accuracy, $acc(q)$, as the fraction of candidate answers matching this pseudo-label (Equation 10). This $acc(q)$ metric serves as a proxy for the problem's difficulty for the current model. We only proceed with synthesis if this accuracy falls within a predefined range $[\tau_{low}, \tau_{high}]$, ensuring the model focuses on problems that are neither trivial nor intractable. **Step 2: Quality Control (on Synthesized Query q')**: To prevent the introduction of malformed or unsolvable problems, we apply a simple but effective heuristic: each newly synthesized query q' must not exceed a maximum token length L_{max}. This filters out incomplete or overly complex generations. The effectiveness of our entire filtering mechanism is verified by a series of ablation studies in terms of filtering thresholds τ_{low}, τ_{high}, and L_{max}.

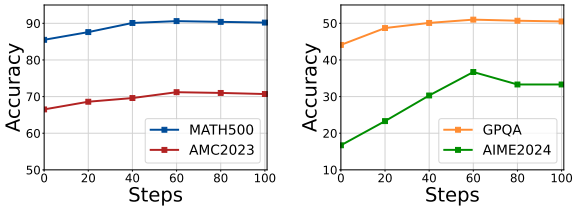
A.2 Ablation Study for Hyperparameters

We first investigate the sensitivity to filtering thresholds τ_{low} and τ_{high} on MATH500 dataset. Table 5 shows that our chosen thresholds ($\tau_{low} = 0.125$, $\tau_{high} = 0.875$) achieve optimal performance. We



(a) MATH500 and AMC2023. (b) GPQA and AIME2024.

Figure 5: Hyperparameter analysis of the warmup steps for intra-group exploration (E_{intra}) on various reasoning datasets..



(a) MATH500 and AMC2023. (b) GPQA and AIME2024.

Figure 6: Hyperparameter analysis of the warmup steps cross-group exploration (E_{cross}) on various reasoning datasets.

have also conducted an additional ablation study on L_{max} to further validate our quality control step. The results in Table 6 show that $L_{max} = 1024$ provides the best balance, effectively filtering out malformed queries without being overly restrictive. The stable performance around this value further demonstrates the robustness of our filtering approach. Furthermore, our TTVS incorporates other critical hyperparameters: the number of warm-up steps for intra-group exploration (E_{intra}) and for cross-group exploration (E_{cross}). The former determines the transition point from exploration to accuracy-driven exploitation, while the latter controls the onset of consistency-driven exploration. To identify the optimal values for these parameters, we conducted a series of sensitivity analyses. First, we evaluated the impact of E_{intra} on model performance across a range of values: 0, 20, 40, 60, 80, 100. As illustrated in Figures 5a and 5b, performance generally peaked when E_{intra} was set to 40 across most reasoning tasks, including AMC2023, GPQA, and AIME2024. Subsequently, we investigated the effect of E_{cross} in a similar manner. The results, shown in Figures 6a and 6b, indicate that the TTVS framework achieved optimal or near-optimal performance on all benchmarks when E_{cross} was set to 60. Consequently, we adopted default values of 40 and 60 for E_{intra} and E_{cross} in

Methods	MATH500	AMC2023	GPQA
Qwen3-4B	64.0	26.5	26.3
TTRL (32 Rollouts)	85.0	61.4	43.0
TTRL (64 Rollouts)	86.9	65.5	45.0
TTVS (32 Rollouts)	90.3	71.1	48.0

Table 7: Performance comparison with increased rollouts for TTRL

Methods	Qwen3-4B	Qwen2.5-Instruct-3B
TTVS (16 Rollouts)	88.0	73.8
TTVS (32 Rollouts)	90.3	75.0
TTVS (64 Rollouts)	92.9	76.2

Table 8: Ablation study for the number of rollouts for TTVS.

Methods	LLaMA-3.2-3B-Instruct	Qwen3-4B
Init	44.1	53.2
TTRL	46.6	62.8
TTVS	49.0	65.9

Table 9: Results on DeepMath-103K subset.

Methods	LLaMA-3.2-3B-Instruct	Qwen3-4B
Init	49.8	56.6
TTRL	51.3	58.1
TTVS	52.1	59.1

Table 10: Results on MedMCQA.

all experiments unless otherwise specified. We also analyzed the sensitivity to our difficulty filtering thresholds.

A.3 Ablation Study for Rollouts

We first conducted an ablation study where we doubled the computational budget for TTRL by increasing its rollouts to 64. As shown in Table 7, simply increasing computation for TTRL yields only marginal performance gains (2%-4%), demonstrating diminishing returns on a static dataset. More importantly, our more compute-efficient TTVS (with 32 rollouts) still significantly outperforms the more expensive TTRL (with 64 rollouts). Furthermore, we investigated the impact of varying the number of rollouts for TTVS. As shown in Table 8, TTVS effectively improves performance with more variants. This indicates that our chosen value offers a strong balance between performance and efficiency, and the method is not overly sensitive to this parameter.

A.4 Clarification for Trade-off Structure

To analyze computational cost, we provide a detailed methodology clarification demonstrating that TTVS's performance gains stem from its novel mechanism, not from a higher computational budget. TTVS was designed for efficiency and does not use k times more rollouts in parallel. For each optimization step of the policy model, the number of rollouts used is identical to the baseline TTRL (i.e., 32 rollouts). TTVS sequentially utilizes rollouts from different data sources (original, synthetic, mixed) to improve the quality and diversity of the training signal, rather than increasing the quantity of parallel computation.

A.5 Evaluation on More Unseen Dataset

To address the potential for data contamination, we conducted performance analysis in Table 9 on a subset of the recently released DeepMath-103K dataset (He et al., 2025), which is unseen by our models. We created a representative subset by randomly sampling 20 question-answer pairs from each difficulty level. TTVS consistently outperforms both the initial model and the TTRL baseline, demonstrating that its gains are genuine and not an artifact of data leakage.

A.6 Evaluation on Domain-Specific Dataset

To investigate the practical value of TTVS in domains with scarce annotated data, we evaluated it on the well-known medical benchmark, MedMCQA (Pal et al., 2022). As shown in Table 10 TTVS again shows a clear and consistent advantage over TTRL, confirming its effectiveness in specialized domains.

A.7 Performance on Base Models without Instruction-following Ability

To demonstrate that the benefits of TTVS are not limited to instruction-following models, we conducted an experiment on the Qwen2.5-Math-1.5B (Base model). To ensure this condition is met even for base models with initial limitations, we introduced an effective expert-level model warmup phase. For the first 100 training steps, we use a small set of high-quality synthetic data generated by an expert model (DeepSeek R1). It is designed to efficiently bootstrap the base model's specific rewriting skills, enabling it to fully participate in the main TTVS self-improvement loop. Our results in Table 11 demonstrate that with this minimal

Methods	MATH500	AMC2023	GPQA
Qwen2.5-Math-1.5	32.7	28.6	24.9
TTRL	70.7	48.9	26.7
TTVS	76.0	50.4	28.6

Table 11: Performance Analysis on Base Model.

Methods	LLaMA-3.2-3B-Instruct	Qwen3-4B	Qwen2.5-Instruct-3B
Init	65.5	82.6	77.2
TTRL	71.9	87.5	78.9
TTVS	73.3	90.7	81.2

Table 12: Performance Analysis on OpenBookQA.

warmup, TTVS significantly outperforms TTRL on the same base model. This confirms that the core benefits of TTVS—harnessing dynamic data synthesis for self-improvement—are not restricted to models with strong pre-existing capabilities. Our warmup strategy provides a simple yet powerful mechanism to unlock the potential of TTVS on a broader range of models, showcasing the versatility of our framework.

A.8 Applicability to Open-Ended Tasks

To explore the boundaries of our method's applicability beyond purely mathematical reasoning, we conducted an additional experiment on OpenBookQA in Table 12. This benchmark assesses a model's ability to reason by combining given facts with commonsense knowledge, representing a step towards more open-ended reasoning. TTVS maintains a consistent and significant performance advantage over TTRL even in this commonsense reasoning context, suggesting that the benefits of our approach are not strictly limited to mathematical logic.

A.9 Discussion for "Non-verifiable" Scenarios

While the previous implementation of TTVS primarily focuses on domains with objective, verifiable rewards (e.g., mathematics), we acknowledge the challenge of extending this framework to open-ended tasks such as creative writing or summarization. In these "non-verifiable" scenarios, the absence of a rule-based or ground-truth reward signal complicates both the pseudo-labeling and the online filtering processes. To bridge this gap, a promising direction involves transitioning from rule-based verifiers to model-based supervision, specifically through an LM-as-a-Judge paradigm. In this setup, a frozen, superior model (e.g., GPT-4o or a specialized Reward Model) could serve as the "verifiable signal" by evaluating the semantic consistency of synthesized query variants and the quality of reasoning rollouts. Specifically, the "Judge" can verify whether a synthesized creative prompt remains semantically equivalent to the original and use consensus-based scoring or pairwise comparisons to generate the necessary reward signals for self-exploration. Although this introduces considerations regarding the judge's bias and additional API costs, it provides a viable pathway to unlock the self-evolution capabilities of TTVS in broader, more subjective linguistic domains.

A.10 Qualitative Synthesis Examples

To evaluate the robustness of our synthesis engine, we analyze representative samples of generated mathematical problems.

- **Original:** Let $f(x) = |x - p| + |x - 15| + |x - p - 15|$, where $0 < p < 15$. Determine the minimum value taken by $f(x)$ for x in the interval $p \leq x \leq 15$. **(Ans: 15)**
- **Good Variant 1:** What is the minimum value of the function $f(x) = |x - q| + |x - 15| + |x - q - 15|$ for $0 < q < 15$? **(Ans: 15)**
- **Good Variant 2:** Let $g(x) = |x - m| + |x - 15| + |x - m - 15|$, where $0 < m < 15$. Determine the minimum value taken by $g(x)$ for x in the interval $m \leq x \leq 15$. **(Ans: 15)**
- **Failure Variant:** Let $f(x) = |x - 3| + |x - 7| + |x - 10|$. Determine the value of x in the interval $[3, 7]$ that minimizes $f(x)$. **(Ans: 7)**

The synthesized outputs demonstrate the model's proficiency in structural preservation and logical migration. In the "Good Variants," the model successfully executes variable renaming (e.g., $p \to q, m$) and maintains the delicate relationship between the parameters and the interval constraints. Meanwhile, these failures are relatively rare and they typically stem from minor numerical hallucinations or logic inversions. This suggests that the model is not merely performing surface-level text substitution but is capturing the underlying functional constraints required for the problem to remain well-posed.

A.11 Performance Comparison (Avg@16 / Maj@16)

To eliminate the variance inherent in stochastic sampling, we employ ensemble metrics—$Avg@16$

and $Maj@16$. The data reveals a consistent and significant performance gain; TTVS does not merely rely on "lucky" samples but fundamentally shifts the distribution of reasoning trajectories toward higher correctness. Notably, TTVS yields an additional +5.3% (MATH500) and +9.4% (AMC2023) over the strong Maj@16 baseline. These results confirm that our strategy is not just complementary to inference-time scaling but fundamentally superior in cultivating core reasoning capabilities.

Metric	Method	MATH500	AIME2024	AMC2023	GPQA
Avg@16	TTRL	85.0	26.7	61.4	43.0
Avg@16	**TTVS (Ours)**	**90.3**	**36.7**	**71.1**	**48.9**
Maj@16	TTRL	86.4	26.7	62.1	43.2
Maj@16	**TTVS (Ours)**	**91.7**	**36.7**	**71.5**	**49.1**

Table 13: Performance Comparison on Avg@16 and Maj@16 metrics.