TTSR: Test-Time Self-Evolving via Reflection

Anonymous ACL submission

Abstract

Test-time training (TTT) adapts large language models (LLMs) during inference using only unlabeled test inputs. Existing methods, however, face two major bottlenecks on hard reasoning tasks: (1) *lack of learnable samples*, as self-generated pseudo-labels on difficult questions are often noisy and yield unstable rewards; and (2) *inefficient exploration*, as performance gains depend on repeatedly sampling many rollouts without explicit diagnosis of why previous attempts fail. We propose **TTSR** (**T**est-**T**ime **S**elf-**R**eflection), a self-evolving framework based on a *reflect-then-synthesize* paradigm. A single pretrained model alternates between a *Student* role and a *Teacher* role: the Student solves test questions and updates, while the Teacher analyzes failed trajectories and synthesizes targeted variant questions closer to the Student's capability frontier. TTSR further maintains a cross-iteration *weakness memory* and compiles persistent weaknesses into a lightweight *strategy note* prepended to subsequent Student inputs, so diagnostic knowledge can guide exploration and gradually fade as weaknesses are resolved. Experiments on challenging mathematical reasoning benchmarks show consistent test-time improvements, strong cross-backbone generalization, and transfer to general-domain reasoning tasks. Our code and experimental details are available in the anonymous repository: `https://anonymous.4open.science/r/TTSR-1680`.

1 Introduction

Test-time training (TTT) updates model parameters at inference time using only unlabeled test data (Akyürek et al., 2024; Sun et al., 2020), requiring no ground-truth supervision. Recent formulations cast TTT as test-time reinforcement learning, where rewards are derived from the model's own sampled solutions via self-consistency or outcome verification (Zuo et al., 2025; Yuksekgonul et al., 2026). In practice, however, these methods produce limited and unstable gains on hard reasoning tasks (Huang et al., 2023b; Akyürek et al., 2024). We focus on two key limitations.

Lack of learnable samples. Test questions in hard reasoning benchmarks often sit near or beyond what the model can reliably solve. Majority-voting pseudo-labels then become noisy, and the resulting rewards too unreliable for stable parameter updates (Akyürek et al., 2024; Zuo et al., 2025). In effect, only a narrow band of questions—neither trivially easy nor hopelessly hard—yields useful gradients, leaving the model with few effective training samples at test time (Bae et al., 2025).

Inefficient exploration via repeated rollouts. In part because of this scarcity, current TTT methods sample large numbers of reasoning trajectories, hoping that some will land on correct solutions (Snell et al., 2024). This is expensive and undirected: the model has no mechanism to diagnose *why* earlier attempts failed or to steer later sampling toward its actual weaknesses.

Two observations point toward a different approach. First, *diagnosing where a reasoning trace went wrong is typically easier than producing a correct solution from scratch*, since error localization operates on a given trace rather than requiring one to be constructed (Wang et al., 2022; Huang et al., 2023b). Second, curriculum learning shows that training on calibrated intermediate problems improves accuracy on harder target tasks (Bengio et al., 2009; Wang et al., 2021). Together, these observations motivate a *reflect-then-synthesize* strategy: the model first diagnoses its reasoning failures, then uses the diagnoses to synthesize variant questions near the model's capability frontier as stepping stones.

We propose **TTSR** (**T**est-**T**ime **S**elf-**R**eflection), a test-time self-evolving framework in which a *single* pretrained model alternates between two roles: a *Student* and a *Teacher*. The Student samples

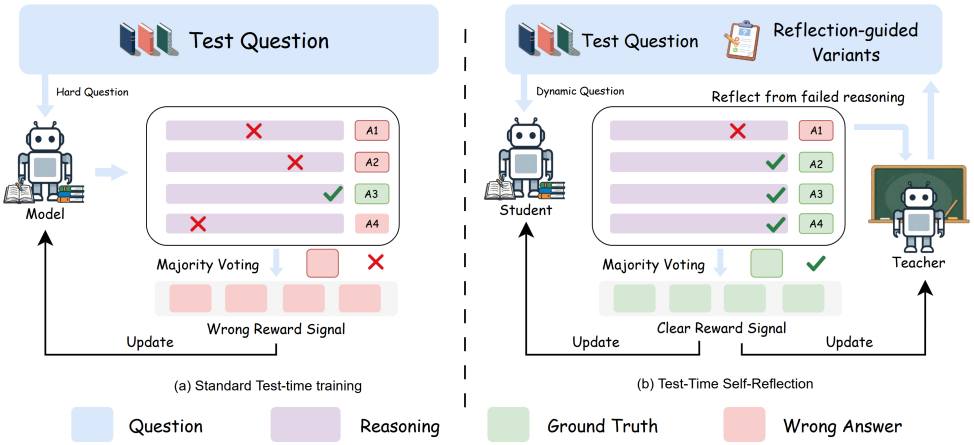

Figure 1: Comparison between standard Test-time training and TTSR. Standard test-time training produces noisy reward signals when test questions are too hard for the model. In contrast, TTSR reflects on failed reasoning to generate moderately difficult variants matched to the model's capability, yielding clearer rewards and more stable test-time updates.

reasoning trajectories for test questions and updates its policy online via GRPO (Shao et al., 2024) with self-supervised rewards. The Teacher analyzes the Student's failed trajectories, extracts structured weakness descriptors through lightweight reflection, and synthesizes variant questions that target these weaknesses. The variants sit closer to the Student's capability frontier and thus provide more informative training signals than the original test questions. Per-iteration reflections are ephemeral: without persistent tracking, the same weaknesses may surface repeatedly. TTSR therefore maintains a cross-iteration *weakness memory* and compiles the most persistent entries into a *strategy note*—concise reasoning guidelines prepended to Student inputs that directly steer subsequent exploration toward unresolved weaknesses. The note fades automatically as weaknesses are resolved through continued adaptation.

We evaluate TTSR on eight mathematical and general reasoning benchmarks across three model backbones. TTSR consistently outperforms test-time RL baselines, achieving up to $+15.4$ average points over the base model, and transfers to general-domain reasoning tasks without domain-specific tuning. These results suggest that *self-reflection with cross-iteration knowledge accumulation* provides a practical pathway for continual reasoning improvement during inference.

Contributions. Our main contributions are:

- We identify the lack of learnable samples and inefficient rollout-based exploration as two key bottlenecks of test-time training, and propose a *reflect-then-synthesize* strategy that leverages failure diagnosis to construct targeted curricula near the model's capability frontier.
- We introduce **TTSR**, a test-time self-evolving framework in which a single model alternates between Student and Teacher roles. TTSR further incorporates a cross-iteration *weakness memory* and a lightweight *strategy note* that persistently carry forward diagnostic knowledge across adaptation iterations.
- We demonstrate that TTSR consistently improves performance on challenging mathematical and general reasoning benchmarks and generalizes across model backbones.

2 Related Work

Self-Play and Curriculum Learning for LLMs (mainly training-time). Self-play has become a common recipe for autonomous improvement in LLMs, from iterative self-training and alignment-oriented loops to role-specialized or co-evolutionary variants (Huang et al., 2023a; Wang et al., 2023; Chen et al., 2024; Lin et al., 2025; Chen et al., 2025; Fang et al., 2025). Recent capability-oriented systems further emphasize adversarial generation, dynamic self-challenging, and data-free evolution (Liu et al., 2025; Xu et al., 2025; Wang et al., 2025; Kuba et al., 2025; Zhou et al., 2025; Wei et al., 2025; Huang et al., 2025; Zhao et al., 2025; He et al., 2025). In parallel, curriculum learning establishes that calibrated intermediate

tasks can improve learning efficiency (Bengio et al., 2009; Wang et al., 2021). However, most of this line is designed for training/post-training regimes with larger budgets and does not explicitly target per-instance failure diagnosis during inference.

Test-Time Training and Test-Time RL. Test-time training (TTT) adapts model parameters at inference time using unlabeled test inputs (Sun et al., 2020; Akyürek et al., 2024). TTRL-style methods cast this as test-time reinforcement learning with self-generated rewards from voting or verification (Zuo et al., 2025; Yuksekgonul et al., 2026). These methods show that online adaptation is feasible, but two challenges remain on hard reasoning benchmarks: pseudo-label reliability degrades when questions are far beyond current capability, and large rollout budgets are often used as a coarse exploration mechanism (Snell et al., 2024; Bae et al., 2025).

Positioning of TTSR. TTSR is a failure-conditioned curriculum adaptation mechanism for test-time training. Its first contribution is that we introduce self-generated curriculum construction into the test-time stage, where adaptation must be online and label-free, while difficult test inputs often lead to too few learnable samples and unstable updates in TTRL. Its second contribution is a *reflect-then-synthesize* loop that converts failed reasoning traces into targeted variant questions near the model's current capability frontier, improving synthesis and exploration efficiency.

3 TTSR: Test-Time Self-Reflection

We present **TTSR**, a test-time self-evolving framework built on GRPO and TTT. A single pretrained model π_θ alternates between two roles at test time: a *Student* that solves test questions and updates its policy via self-supervised rewards, and a *Teacher* that analyzes failed reasoning trajectories, extracts structured weakness descriptors, and synthesizes targeted variant questions. To accumulate diagnostic knowledge across iterations, TTSR maintains a *weakness memory* $\mathcal{M}_t$ and compiles persistent entries into a *strategy note* h_t—concise reasoning guidelines prepended to Student inputs. The overall pipeline is illustrated in Figure 2.

Student Role. Given a test question $x \in \mathcal{X}_{\text{test}}$ and the current strategy note h_t (Section 3.3), the Student samples G reasoning trajectories $\{y_i\}_{i=1}^{G} \sim \pi_\theta(\cdot \mid h_t,\, x)$, each consisting of a chain-of-thought process and a final answer. We write $a(y_i)$ for the final answer extracted from trajectory y_i. These trajectories serve both as inference outputs and as the basis for self-supervised learning signals.

Teacher Role. The Teacher does not solve the test question directly. Instead, it identifies failures in the Student's trajectories relative to a pseudo-target obtained via majority voting, reflects on the underlying reasoning weaknesses, and synthesizes targeted easier variant questions designed to expose and correct them.

3.1 Student: Solving Problems and Test-Time Adaptation

Test-Time Training Set Construction. At test-time iteration t, the Student updates its policy π_{θ_t} using a dynamically constructed training set. Specifically, we define the training dataset at iteration t as

$$\mathcal{D}_t = \mathcal{X}_{\text{test}} \;\cup\; \mathcal{X}_{\text{var}}^{(t-1)}, \tag{1}$$

where $\mathcal{X}_{\text{test}}$ denotes the original test set, and $\mathcal{X}_{\text{var}}^{(t-1)}$ represents the set of variant questions synthesized by the Teacher using the previous policy $\pi_{\theta_{t-1}}$. The Student samples training questions from $\mathcal{D}_t$ and performs test-time adaptation via reward-based optimization.

Majority Voting Reward. For each training question $x \in \mathcal{D}_t$, the Student samples a group of G reasoning trajectories using the current policy:

$$\{y_i\}_{i=1}^{G} \sim \pi_{\theta_t}(\cdot \mid h_t,\, x). \tag{2}$$

Following the Test-Time Reinforcement Learning paradigm, we aggregate the sampled trajectories via majority voting over their final answers to obtain a consensus answer $\hat{a}(x)$, which serves as a pseudo-correct reference. An outcome-level pseudo-correctness reward is then assigned to each trajectory according to whether its final answer agrees with the consensus:

$$R_S(y_i \mid x) = \mathbb{I}[a(y_i) = \hat{a}(x)]\,. \tag{3}$$

Based on these rewards, we define the empirical pseudo-correctness score for question x as

$$s_t(x) = \frac{1}{G}\sum_{i=1}^{G} R_S(y_i \mid x), \tag{4}$$

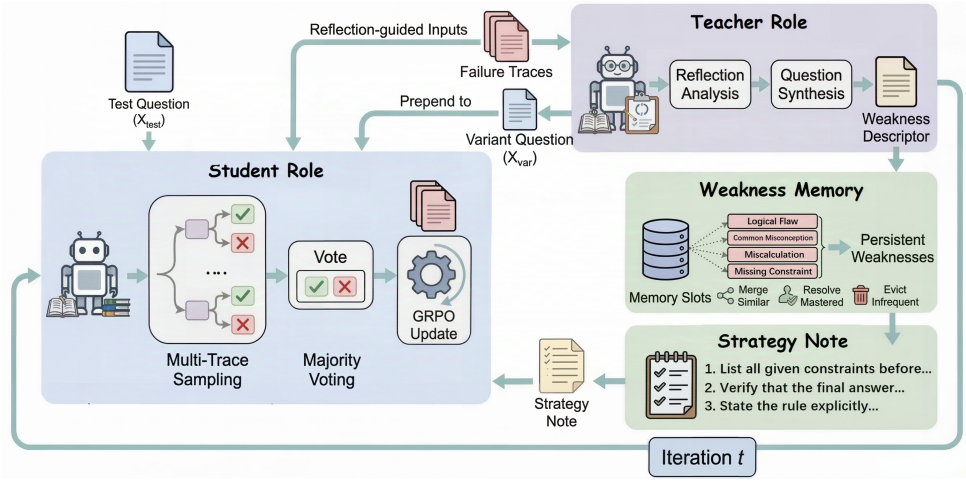

Figure 2: **Overview of the TTSR pipeline.** At each iteration t, the Student samples multiple reasoning trajectories and updates its policy via GRPO with majority-voting pseudo-labels. The Teacher analyzes failed traces to extract structured weakness descriptors, which are accumulated in a cross-iteration *weakness memory*. Persistent weaknesses are compiled into a *strategy note* that is prepended to Student inputs, providing immediate reasoning guidance. The Teacher also synthesizes targeted variant questions conditioned on both current failures and the accumulated memory, forming a closed-loop self-evolving cycle.

which measures the stability of the Student's predictions under stochastic sampling at iteration t. The Student is optimized using GRPO with these pseudo-correctness rewards, enabling stable and label-free test-time adaptation.

3.2 Teacher: Reflection-Guided Curriculum Synthesis

Reflection on Reasoning Steps. At test-time iteration t, the Teacher performs reflection based on the Student's failed reasoning behaviors observed in the previous iteration. Specifically, using the policy $\pi_{\theta_{t-1}}$, we collect a set of reasoning trajectories whose pseudo-correctness rewards are zero. From these trajectories, we randomly sample a subset of failed instances for reflection.

Each sampled instance consists of a tuple $(x, y, \hat{a})$, where x denotes the question, y the Student's reasoning trajectory, and $\hat{a}$ the corresponding pseudo-correct answer obtained via majority voting over final answers. Conditioned on this set of failed tuples, the Teacher with policy π_{θ_t} analyzes the reasoning steps in y and produces a reflection summary that characterizes both the incorrect reasoning patterns and the missing or insufficient steps leading to failure. This reflection focuses on the process-level deficiencies of the Student's reasoning rather than the final answer correctness. Each reflection output is a structured weakness descriptor consisting of a natural-language weakness summary, trigger conditions, and failure signatures. These descriptors serve dual purposes: guiding question synthesis within the current iteration and updating the cross-iteration weakness memory $\mathcal{M}_t$ (Section 3.3).

Reflection-Guided Question Synthesis. Given the extracted weakness descriptors and the accumulated weakness memory, the Teacher synthesizes variant questions that form an adaptive curriculum for the Student. Let $\mathcal{F}^{(t-1)} = \{(x_k, y_k, \hat{a}_k)\}_{k=1}^{|\mathcal{F}|}$ denote the set of failed instances from the previous iteration. We construct a synthesis prompt p_t from both $\mathcal{F}^{(t-1)}$ and the most persistent entries in $\mathcal{M}_t$, then sample M variants:

$$\mathcal{X}_{\text{var}}^{(t)} = \{x'_j\}_{j=1}^{M} \sim \pi_{\theta_t}\Big(\cdot \,\Big|\, p_t,\, \mathcal{F}^{(t-1)},\, \mathcal{M}_t\Big)\,. \tag{5}$$

Each variant preserves the core reasoning structure of the original question while modifying conditions to expose the identified weaknesses. We use group sampling to encourage diversity, so the curriculum covers multiple manifestations of the same deficiency.

Difficulty Reward. We evaluate the difficulty of a variant question x' using the Student's pseudo-correctness score at iteration t. Concretely, the Student samples G trajectories under π_{θ_t} and computes the empirical pseudo-correctness score $s_t(x') \in [0, 1]$ as defined in Section 3.1.

Following prior work showing that variance-based rewards effectively induce capability-

centered curricula in self-play (Liu et al., 2025), we adopt a normalized variance-based frontier reward:

$$R_{\text{diff}}(x') = 4\, s_t(x')\big(1 - s_t(x')\big). \tag{6}$$

This reward corresponds to the normalized variance of a Bernoulli variable with success probability $s_t(x')$ and is maximized at $s_t(x') = 0.5$, i.e., when the Student operates near its capability frontier. It therefore encourages the Teacher to synthesize variant questions that are neither trivial nor degenerate. Further theoretical analysis is provided in Appendix C.3.

Similarity Penalty. To reduce redundancy among synthesized variants, we penalize near-duplicate generations. Let $\mathcal{Z} = \{x'_1, \ldots, x'_M\} \cup \{x\}$ be the set of variants together with the original question. For each variant x'_i, the penalty is

$$R_{\text{sim}}(x'_i, x) = \frac{1}{|\mathcal{Z}| - 1} \sum_{z \in \mathcal{Z} \setminus \{x'_i\}} \max\big(0,\ \text{sim}(x'_i, z) - \tau\big)\,. \tag{7}$$

where $\text{sim}(\cdot,\cdot) \in [0,1]$ is a token-level sequence similarity metric (Appendix C.2) and τ controls the overlap tolerance.

Teacher Reward. For each variant x'_i, the Teacher's overall reward combines difficulty and diversity:

$$R_T(x'_i) = \max\big(0,\ R_{\text{diff}}(x'_i) - \lambda\, R_{\text{sim}}(x'_i, x)\big)\,. \tag{8}$$

Outputs that violate the predefined <question> format are discarded and receive zero reward.

3.3 Cross-Iteration Weakness Memory and Strategy Note

Per-iteration reflections are ephemeral and do not accumulate knowledge across iterations. To persistently track reasoning deficiencies, TTSR maintains a *weakness memory* $\mathcal{M}_t$ and compiles persistent entries into a *strategy note* h_t.

Weakness Memory. At each iteration t, the Teacher's reflection produces weakness descriptors $\mathcal{W}^{(t)} = \{w_1^{(t)}, \ldots, w_n^{(t)}\}$ (Section 3.2). The memory $\mathcal{M}_t$ is updated via three operations:

Merge. If a new descriptor $w' \in \mathcal{W}^{(t)}$ matches an existing entry $w \in \mathcal{M}_{t-1}$ ($\text{sim}(w', w) > \tau_w$, using the same metric as in Section 3.2), we merge w' into w by incrementing its persistence counter $c(w) \leftarrow c(w) + 1$ and updating $w.\text{last_seen} \leftarrow t$; otherwise w' is added as a new entry with $c(w') = 1$.

Resolve. Entries not observed for Δ consecutive iterations ($t - w.\text{last_seen} > \Delta$) are removed, indicating the weakness has been addressed.

Evict. If $|\mathcal{M}_t| > K$ after merging, entries with the lowest persistence count are dropped.

The complete update is:

$$\begin{aligned} \widetilde{\mathcal{M}}_{t+1} &= \text{Resolve}\big(\text{Merge}(\mathcal{M}_t,\ \mathcal{W}^{(t)}),\ \Delta\big), \\ \mathcal{M}_{t+1} &= \text{Evict}\big(\widetilde{\mathcal{M}}_{t+1},\ K\big). \end{aligned} \tag{9}$$

Strategy Note. At the beginning of each iteration, we select the top-N entries from $\mathcal{M}_t$ ranked by persistence count and compile them into a strategy note:

$$h_t = \text{Compile}(\mathcal{M}_t^{\text{top-}N}). \tag{10}$$

The note is prepended to all Student inputs during both sampling and GRPO training. As the Student improves, resolved weaknesses are removed from $\mathcal{M}_t$ and the corresponding guidelines automatically drop from h_t, so the note serves as a transient scaffold that fades as the underlying capability catches up.

4 Experiment

4.1 Experimental Setting

Models and Baselines. We evaluate on three base pretrained language models: **Qwen3-4B-Base**, **Qwen3-8B-Base** (Yang et al., 2025), and **OctoThinker-8B-Hybrid-Base** (Wang et al., 2025). We compare against two test-time optimization baselines: **TTRL** (Zuo et al., 2025), which adapts the model on unlabeled test instances using self-generated reinforcement signals, and **R-Zero** (Huang et al., 2025), which enables data-free self-improvement through reinforcement learning. The **Base model** performs direct inference without parameter updates. We additionally report a **Self-Consistency** (SC) (Wang et al., 2022) baseline that applies majority voting over $G = 16$ sampled trajectories without any parameter updates, to disentangle the effect of inference-time scaling from test-time training. All baselines are evaluated under the same test-time setting for fair comparison.

Datasets and Evaluation. Mathematical Reasoning. We evaluate on **AMC23**,

MATH-500 (Hendrycks et al., 2021), **Minerva** (Lewkowycz et al., 2022), **OlympiadBench** (He et al., 2024), **AIME 2024**, and **AIME 2025**, which cover a wide range of difficulty levels and require multi-step symbolic manipulation. **General Reasoning.** We additionally evaluate on **GPQA-Diamond** (Rein et al., 2024) and **MMLU-Pro** (Wang et al., 2024) to assess reasoning ability beyond mathematics. Each benchmark is adapted independently.

Training Configuration. The Teacher generates $M = 2$ variant questions per original test question per iteration; the Student samples $G = 16$ reasoning trajectories per question. The policy is optimized via GRPO with pseudo-correctness rewards. All hyperparameters—including similarity penalty ($\lambda = 1.0$, $\tau = 0.75$), weakness memory ($K = 10$, $\Delta = 3$, $\tau_w = 0.6$), and strategy note ($N = 3$)—are fixed across all models and benchmarks; see Appendix B.2 for full details.

4.2 Main Results

Table 1 summarizes results across all benchmarks.

TTSR achieves consistent improvements across models and domains. On Qwen3-4B-Base, TTSR raises average accuracy by $+11.7$ points over the base model. On Qwen3-8B-Base, the average improvement is $+12.0$ points, and on OctoThinker-8B-Hybrid-Base—a model with reasoning-oriented mid-training (Wang et al., 2025)—the gains reach $+15.4$ points. Beyond mathematics, TTSR also improves general reasoning: gains on GPQA-Diamond and MMLU-Pro range from 3 to 7 points over the strongest baseline, indicating that the reflection-guided curriculum transfers across domains.

TTSR outperforms existing test-time training methods on average and on most benchmarks. Compared to TTRL and R-Zero, TTSR delivers clear average gains across model backbones, with improvements on the majority of benchmarks. On Qwen3-8B-Base, TTSR surpasses TTRL by $+4.2$ points on average; on OctoThinker-8B, the margin widens to $+6.8$. This gap indicates that reflection-guided variant synthesis provides more informative training signal than direct adaptation on raw test questions.

TTSR shows the largest improvements on challenging benchmarks. The advantage is most pronounced on AIME 2024/2025, where the original questions lie far beyond the model's reliable solving range. On Qwen3-4B-Base, TTSR exceeds TTRL by $+8.0$ on AIME24 and $+2.9$ on AIME25. On Qwen3-8B-Base, TTSR reaches 30.8 on AIME24 ($+4.7$ over TTRL) and 19.1 on AIME25 ($+3.4$). When TTRL's pseudo-labels become unreliable on intractable problems, Teacher-synthesized variants bridge the difficulty gap by creating frontier-level stepping stones that progressively approach the target difficulty.

We additionally compare against Self-Consistency (SC), which applies majority voting over $G = 16$ trajectories without parameter updates; full results are reported in Table 12 (Appendix H). All test-time training methods substantially outperform SC, confirming that TTSR 's gains stem from parameter adaptation rather than inference-time scaling. Multi-seed experiments on Qwen3-4B-Base (Appendix K) further confirm that improvements are stable across runs.

4.3 Analysis

We conduct targeted analyses to validate the key design choices of TTSR.

Q1: Do synthesized variants concentrate near the capability frontier? A key hypothesis is that reflection-guided synthesis generates questions near the Student's capability frontier, where $R_{\text{diff}} = 4s_t(1 - s_t)$ is high. On Qwen3-8B-Base, we compare pseudo-correctness $s_t(x')$ across three sources: (i) original test questions, (ii) variants without reflection (random structural variation), and (iii) full TTSR variants. We report the *frontier ratio*: the fraction with $s_t \in [0.2, 0.8]$, where R_{diff} is at least 64% of its maximum (Table 10, Appendix E). On MATH500, the frontier ratio rises from 16.2% (original) to 67.4% (TTSR), a $4.2\times$ increase. On AIME25, where original questions have a frontier ratio of only 8.3%, TTSR variants achieve 64.2%, demonstrating that reflection is especially valuable when the test set lies far from the model's capability boundary. Removing reflection reduces the frontier ratio to 43.8% and 35.6%, indicating that weakness-aware diagnosis provides gains beyond what random structural variation can achieve.

Q2: Is co-evolution necessary, or can a stronger static teacher suffice? The ablation study (Table 13, Appendix I) shows that removing Teacher test-time updates—equivalent to using a frozen Teacher at iteration 0—causes the largest overall

Method	AMC23	MATH500	Minerva	Olympiad	AIME24	AIME25	GPQA-D	MMLU-Pro	Δ
Qwen3-4B-Base									
Base Model	45.5	72.0	32.4	40.2	12.4	5.8	25.7	52.0	–
R-Zero	54.0	76.8	40.7	44.0	18.2	9.1	28.4	55.0	+5.0
TTRL	55.5	79.0	43.6	46.0	17.6	14.7	29.1	56.0	+6.9
TTSR (ours)	**61.0**	**82.4**	**53.0**	**45.3**	**25.6**	**17.6**	**34.2**	**60.8**	**+11.7**
Qwen3-8B-Base									
Base Model	51.5	77.8	39.6	41.2	15.9	9.8	33.1	58.6	–
R-Zero	58.5	82.0	47.8	48.6	22.6	13.4	36.7	61.5	+5.5
TTRL	61.5	84.2	50.2	50.8	26.1	15.7	38.4	62.8	+7.8
TTSR (ours)	**66.5**	**87.4**	**54.9**	**55.2**	**30.8**	**19.1**	**42.6**	**66.7**	**+12.0**
OctoThinker-8B-Hybrid-Base									
Base Model	28.5	45.6	15.2	16.8	7.1	3.9	15.2	26.8	–
R-Zero	35.5	54.0	22.9	25.4	11.6	6.8	19.7	31.4	+6.0
TTRL	38.5	57.2	26.4	28.7	13.9	8.2	21.5	33.2	+8.6
TTSR (ours)	**46.5**	**64.8**	**34.1**	**36.8**	**19.7**	**12.4**	**27.9**	**39.8**	**+15.4**

Table 1: Comprehensive evaluation across mathematical and general reasoning benchmarks. AIME 2024/2025 report Mean@32 accuracy; all other benchmarks report Pass@1 (greedy decoding). Best results per base model are in **bold**.

drop (up to -5.8 on AIME25 and Olympiad). This confirms that Teacher–Student co-evolution is critical: a co-evolving same-size Teacher keeps the curriculum aligned with the Student's evolving capability frontier, whereas a frozen Teacher produces increasingly mismatched variants as training progresses.

Q3: How does TTSR perform with limited test data? To assess whether TTSR's gains rely on large test sets, we evaluate data efficiency on **AIME24** with **Qwen3-4B-Base**. TTSR consistently outperforms TTRL at all data ratios: 10% (12.3 vs 9.5, +2.8), 20% (14.6 vs 11.2, +3.4), 30% (15.3 vs 11.8, +3.5), and 100% (25.6 vs 17.6, +8.0). These results suggest that reflection-guided variant synthesis amplifies the value of limited test data by constructing additional frontier-difficulty training signals. It also weakens an instance-memorization explanation: even with only a small fraction of test instances, TTSR retains a clear advantage.

We discuss computational overhead and iso-compute comparisons in Appendix J.

Q4: Do improvements transfer across domains? We evaluate cross-benchmark transfer on Qwen3-8B-Base. The OOD experiments (Figure 6) show that TTSR transfers well across mathematical benchmarks: training on Olympiad yields 25.4 on AIME24, exceeding TTRL's 22.0 (+3.4), while the Base model achieves only 15.9. Beyond mathematics, the training dynamics (Figure 4) reveal meaningful cross-domain gains on GPQA-D (33.1 → 36.2, +3.1) and MMLU-Pro (58.6 → 61.2, +2.6), far exceeding TTRL (+1.5 and +1.2, respectively). These cross-domain gains suggest that TTSR acquires transferable reasoning strategies rather than memorizing benchmark-specific patterns, and all baselines share identical access to test data, so comparisons remain fair.

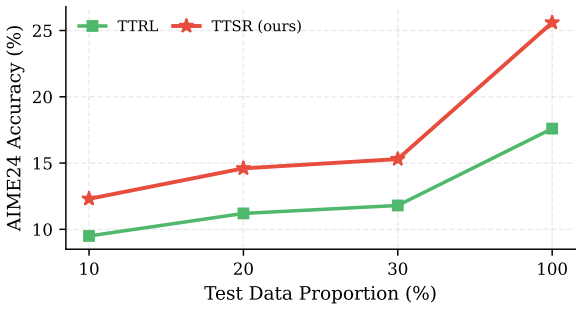

Figure 3: **Data efficiency on AIME24 (Qwen3-4B-Base).** Accuracy under four test-data proportions.

4.4 Training Dynamics

Figure 4 tracks accuracy across test-time iterations on Qwen3-8B-Base. On both MATH500 and AIME25, TTSR starts below TTRL at iteration 2—reflecting the overhead of an initially empty weakness memory—but overtakes TTRL once the memory accumulates sufficient entries. TTRL shows larger mid-training fluctuations, including a dip on MATH500 at iteration 8 (83.8 → 82.6), whereas TTSR exhibits steadier upward progress. Even when trained exclusively on AIME25, TTSR yields cross-domain gains on GPQA-D (+3.1) and

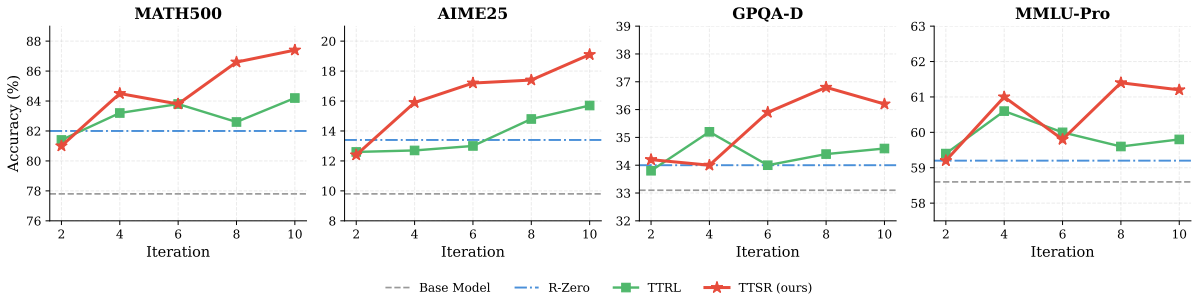

Figure 4: **Training dynamics of TTSR on Qwen3-8B-Base.** Left two panels show *in-domain* accuracy on the training benchmarks (MATH500, AIME25); right two panels show *cross-domain* accuracy on held-out general reasoning benchmarks (GPQA-D, MMLU-Pro) while training on AIME25. Dashed and dash-dotted horizontal lines denote the Base Model and R-Zero baselines, respectively.

MMLU-Pro (+2.6), exceeding TTRL (+1.5 and +1.2), suggesting that the weakness memory captures reasoning strategies that transfer beyond the mathematical domain. Additional training dynamics on AMC23 and Olympiad are provided in Appendix G.

4.5 Ablation Study

Table 2 progressively enriches the variant-synthesis pipeline on Qwen3-8B-Base to isolate the source of TTSR's gains. A full component-level ablation is deferred to Table 13 (Appendix I).

Variant synthesis already helps. Adding difficulty-controlled variant generation with a same-size (8B) teacher and training via GRPO (+ Synthesis) lifts accuracy by +2.7–+6.4 points over the base model, confirming that synthesized curriculum data provides meaningful learning signals beyond the original test set.

Scaling the teacher has diminishing returns. Replacing the 8B teacher with Qwen3-14B (+ Strong Synthesizer) yields an additional +1.6–+3.4 points. While the larger teacher generates higher-quality variants, the gains plateau because, without reflection and co-evolution, the curriculum cannot adapt to the Student's evolving weaknesses.

TTSR's framework matters most. TTSR adds reflection-guided synthesis, Teacher–Student co-evolution, and auxiliary components (difficulty reward, similarity penalty, weakness memory, strategy note) on top of the same-size 8B teacher, achieving a further +2.8–+5.2 points over the 14B synthesizer. The total improvement of TTSR over the base model (+9.3–+14.0) is 1.6× larger than the strong synthesizer's (+4.3–+9.8), demonstrating that TTSR's adaptive curriculum design matters more than raw teacher capacity.

Config.	M500	A25	Oly.	GPQA-D
Base Model	77.8	9.8	41.2	33.1
+ Synthesis	82.2	12.8	47.6	35.8
+ Strong Synth.	84.6	15.0	51.0	37.4
+ TTSR (Ours)	**87.4**	**19.1**	**55.2**	**42.6**

Table 2: Additive analysis on Qwen3-8B-Base. Starting from the base model, we progressively enrich the variant-synthesis pipeline. "Strong Synthesizer" replaces the same-size teacher. Full component-level ablation is in Table 13 (Appendix I).

5 Conclusion

We identify two core bottlenecks of test-time training on hard reasoning tasks: noisy pseudo-labels on overly difficult questions (few learnable samples) and inefficient rollout exploration without failure diagnosis. To address both, we propose **TTSR**, a *reflect-then-synthesize* framework in which one model alternates between Student and Teacher roles. The Teacher analyzes failed trajectories, extracts structured weaknesses, and synthesizes targeted variants near the Student's capability frontier. A cross-iteration weakness memory compiles persistent patterns into a strategy note that guides subsequent reasoning.

Across eight mathematical and general reasoning benchmarks and three base models, TTSR improves average accuracy by +11.7 to +15.4 points over direct inference and consistently outperforms prior test-time training baselines. Ablations show that Teacher–Student co-evolution and reflection-guided synthesis are the primary contributors, and cross-domain results indicate that gains transfer beyond the training distribution.

6 Limitations

In this work, we introduce TTSR, a test-time self-evolving framework that improves large language models on challenging reasoning tasks through reflection-guided curriculum synthesis and cross-iteration weakness accumulation. However, there are still some limitations: (1) Our current evaluation mainly focuses on mathematical and reasoning-intensive benchmarks. Although these settings provide a controlled testbed for studying test-time self-improvement, extending the evaluation to a broader range of task families would further strengthen the generality of the conclusions. (2) To ensure fair comparison across methods, we adopt a largely unified test-time adaptation setup in our experiments. Exploring a wider design space of update frequency, synthesis budget, and iteration schedules may yield additional insights into the behavior of TTSR under different resource conditions. (3) While our experiments include several representative models, the current study is not intended to be an exhaustive comparison across all model families, parameter scales, and alignment settings. A broader model coverage would provide a more complete picture of how TTSR behaves under different model configurations. In future work, we aim to broaden the evaluation of TTSR beyond mathematical and reasoning-intensive benchmarks to more diverse domains and task settings. We are also interested in studying richer adaptation schedules and broader model coverage, which may provide a more comprehensive understanding of how TTSR scales across different resource budgets and model families.

References

Ekin Akyürek, Mehul Damani, Adam Zweiger, Linlu Qiu, Han Guo, Jyothish Pari, Yoon Kim, and Jacob Andreas. 2024. The surprising effectiveness of test-time training for few-shot learning. *arXiv preprint arXiv:2411.07279*.

Sanghwan Bae, Jiwoo Hong, Min Young Lee, Hanbyul Kim, JeongYeon Nam, and Donghyun Kwak. 2025. Online difficulty filtering for reasoning oriented reinforcement learning. *arXiv preprint arXiv:2504.03380*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Jiaqi Chen, Bang Zhang, Ruotian Ma, Peisong Wang, Xiaodan Liang, Zhaopeng Tu, Xiaolong Li, and Kwan-Yee K Wong. 2025. Spc: Evolving self-play critic via adversarial games for llm reasoning. *arXiv preprint arXiv:2504.19162*.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*.

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*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, and 1 others. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850.

Yicheng He, Chengsong Huang, Zongxia Li, Jiaxin Huang, and Yonghui Yang. 2025. Visplay: Self-evolving vision-language models from images. *arXiv preprint arXiv:2511.15661*.

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*.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. 2025. R-zero: Self-evolving reasoning llm from zero data. *arXiv preprint arXiv:2508.05004*.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023a. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore. Association for Computational Linguistics.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023b. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.

Jakub Grudzien Kuba, Mengting Gu, Qi Ma, Yuandong Tian, Vijai Mohan, and Jason Chen. 2025. Language self-play for data-free training. *arXiv preprint arXiv:2509.07414*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, and 1 others. 2022. Solving quantitative reasoning problems with language models.

Advances in neural information processing systems, 35:3843–3857.

Zi Lin, Sheng Shen, Ilia Kulikov, Jingbo Shang, Jason Weston, and Yixin Nie. 2025. Learning to solve and verify: A self-play framework for code and test generation. *arXiv preprint arXiv:2502.14948*.

Bo Liu, Chuanyang Jin, Seungone Kim, Weizhe Yuan, Wenting Zhao, Ilia Kulikov, Xian Li, Sainbayar Sukhbaatar, Jack Lanchantin, and Jason Weston. 2025. Spice: Self-play in corpus environments improves reasoning. *arXiv preprint arXiv:2510.24684*.

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*.

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*.

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*.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. 2020. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pages 9229–9248. PMLR.

Xin Wang, Yudong Chen, and Wenwu Zhu. 2021. A survey on curriculum learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):4555–4576.

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*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 13484–13508.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290.

Zengzhi Wang, Fan Zhou, Xuefeng Li, and Pengfei Liu. 2025. Octothinker: Mid-training incentivizes reinforcement learning scaling. *arXiv preprint arXiv:2506.20512*.

Lai Wei, Yuting Li, Chen Wang, Yue Wang, Linghe Kong, Weiran Huang, and Lichao Sun. 2025. First sft, second rl, third upt: Continual improving multimodal llm reasoning via unsupervised post-training. *arXiv preprint arXiv:2505.22453*.

Fangzhi Xu, Hang Yan, Chang Ma, Haiteng Zhao, Qiushi Sun, Kanzhi Cheng, Junxian He, Jun Liu, and Zhiyong Wu. 2025. Genius: A generalizable and purely unsupervised self-training framework for advanced reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13153–13167.

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*.

Mert Yuksekgonul, Daniel Koceja, Xinhao Li, Federico Bianchi, Jed McCaleb, Xiaolong Wang, Jan Kautz, Yejin Choi, James Zou, Carlos Guestrin, and 1 others. 2026. Learning to discover at test time. *arXiv preprint arXiv:2601.16175*.

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*.

Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. 2025. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*.

Yifei Zhou, Sergey Levine, Jason Weston, Xian Li, and Sainbayar Sukhbaatar. 2025. Self-challenging language model agents. *arXiv preprint arXiv:2506.01716*.

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. *arXiv preprint arXiv:2504.16084*.

A Background: GRPO and TTT

This section briefly reviews the optimization and adaptation techniques underlying our framework, including Group Relative Policy Optimization (GRPO) (Shao et al., 2024) for reward-based learning and Test-Time Training (TTT) (Zuo et al., 2025) for label-free model adaptation at inference time.

Group Relative Policy Optimization (GRPO). Group Relative Policy Optimization (GRPO) (Shao et al., 2024) is a policy-gradient algorithm for large language models that removes the need for an explicit value function. Instead of estimating advantages via a learned critic, GRPO derives relative advantages by normalizing outcome-level rewards across a group of sampled responses, which enables stable optimization under sparse or binary supervision.

We model the language model as a stochastic policy π_θ that generates a response $y = (y_1, \ldots, y_{|y|})$ conditioned on an input question q. For each question q, a behavior policy $\pi_{\theta_{\text{old}}}$ samples a group of G candidate responses $\{y_i\}_{i=1}^G$, each receiving a sequence-level reward R_i from a verifier.

To derive learning signals without a value function, GRPO normalizes rewards within the sampled group and defines a relative advantage for each response:

$$\tilde{A}_i = \frac{R_i - \text{mean}(\mathbf{R})}{\text{std}(\mathbf{R}) + \delta}, \qquad \mathbf{R} = \{R_1, \ldots, R_G\}, \tag{11}$$

where δ is a small constant for numerical stability. The same advantage $\tilde{A}_i$ is shared across all token positions within response y_i.

GRPO updates the policy parameters by maximizing a PPO-style clipped surrogate objective with an explicit KL regularization term:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q\sim\mathcal{D}_q,\, \{y_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \Big(\min\big(r_{i,t}(\theta)\, \tilde{A}_i,\, \text{clip}(r_{i,t}(\theta),\, 1-\epsilon,\, 1+\epsilon)\, \tilde{A}_i\big) - \beta\, \mathbb{D}_{\text{KL}}(\pi_\theta \,\|\, \pi_{\theta_{\text{old}}}) \Big) \right], \tag{12}$$

where

$$r_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t} \mid q, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid q, y_{i,<t})}, \qquad \tilde{A}_i = \frac{R_i - \text{mean}(\{R_i\}_{i=1}^G)}{\text{std}(\{R_i\}_{i=1}^G) + \delta}. \tag{13}$$

Here, ϵ controls the clipping range, and β balances the KL regularization term, which constrains policy updates to remain close to the behavior policy and improves optimization stability.

Test-Time Training (TTT). Test-Time Training enables a pretrained policy to adapt its parameters during inference using only unlabeled test inputs. Let $\mathcal{X}_{\text{test}} = \{x^{(t)}\}$ denote the set of test queries.

Starting from an initial policy $\pi_{\theta_{\text{old}}}$, TTT seeks adapted parameters by optimizing a self-supervised objective:

$$\theta^\dagger = \arg\max_\theta \; \mathbb{E}_{y\sim\pi_\theta(\cdot|x^{(t)})}\big[\mathcal{R}_{\text{TTT}}(y, \tilde{y})\big], \tag{14}$$

where $\tilde{y}$ denotes a pseudo-target constructed from the model's own predictions, typically obtained via self-consistency or consensus-based aggregation. The reward function $\mathcal{R}_{\text{TTT}}$ measures agreement between a candidate response and the pseudo-target, providing a learning signal for inference-time adaptation.

B Experiment Details

B.1 Benchmarks

Mathematical Reasoning Benchmarks.

- **AMC23** (Zeng et al., 2025): Competition-style problems that require precise reasoning and careful case analysis.
- **MATH-500** (Hendrycks et al., 2021): A diverse set of mathematical topics (algebra, geometry, number theory, and combinatorics) used to evaluate structured mathematical reasoning.
- **Minerva** (Lewkowycz et al., 2022) and **OlympiadBench** (He et al., 2024): Advanced olympiad-style problems emphasizing long reasoning chains and non-trivial mathematical insights.
- **AIME 2024** and **AIME 2025**: Highly challenging settings where a single reasoning error can invalidate the entire solution.

General Reasoning Benchmarks.

- **GPQA-Diamond**: Expert-level scientific question answering, requiring deep reasoning over specialized knowledge.
- **MMLU-Pro**: Broad multi-domain reasoning evaluation across a wide range of subjects.

Evaluation Metrics.

- **Mean@32** (AIME 2024/2025): Accuracy under multiple sampled reasoning trajectories.
- **Greedy Decoding (Pass@1)** (others): Greedy decoding accuracy.

B.2 Training Setting

All models and benchmarks share the configuration below unless otherwise noted. At each iteration, the Student is updated on both the original test questions and the Teacher-synthesized variants from the previous iteration.

Similarity Penalty Configuration. To discourage redundant or near-duplicate question generation during reflection-guided synthesis, we incorporate a similarity-based penalty into the Teacher's reward. Pairwise similarities are computed between synthesized variant questions and the original test question, as well as among the synthesized questions within the same batch. The similarity threshold is fixed to $\tau = 0.75$, and the penalty coefficient is set to $\lambda = 1.0$ in all experiments. These values are kept constant across models and benchmarks.

C Method Details

Algorithm	**Batch Size**	**Student Rollout G**	**Teacher Variants M**	**Iterations T**	**KL Coef**	**Learning Rate**	**Max Len**
GRPO	16	16	2	10	0.001	$3e-7$	4096

Table 3: Important Training Parameters for Test-Time Adaptation.

Algorithm 1 TTSR: Test-Time Self-Evolving via Reflection

Require: Test set $\mathcal{X}_{\text{test}}$, initial policy π_{θ_0}, iterations T, Student rollout count G, Teacher variant count M, memory budget K, stale threshold Δ, strategy-note size N

Initialize weakness memory $\mathcal{M}_0 \leftarrow \emptyset$, variant set $\mathcal{X}_{\text{var}}^{(0)} \leftarrow \emptyset$, cached rollout buffer $\mathcal{R}_{\text{var}}^{(0)} \leftarrow \emptyset$
for $t = 1$ to T **do**
  Compile strategy note $h_t \leftarrow \text{Compile}(\mathcal{M}_{t-1}^{\text{top-}N})$
  Construct mixed training set $\mathcal{D}_t \leftarrow \mathcal{X}_{\text{test}} \cup \mathcal{X}_{\text{var}}^{(t-1)}$
  Initialize failed-instance buffer $\mathcal{F}^{(t)} \leftarrow \emptyset$
  for all $x \in \mathcal{D}_t$ **do**
    if $x \in \mathcal{X}_{\text{var}}^{(t-1)}$ **and** cached rollout exists in $\mathcal{R}_{\text{var}}^{(t-1)}$ **then**
      Load cached trajectories $\{y_i\}_{i=1}^{G}$, consensus answer $\hat{a}(x)$, and rewards $\{R_S(y_i \mid x)\}_{i=1}^{G}$
    else
      Sample Student trajectories $\{y_i\}_{i=1}^{G} \sim \pi_{\theta_{t-1}}(\cdot \mid h_t, x)$
      Compute consensus answer $\hat{a}(x) = \text{Maj}(\{a(y_i)\}_{i=1}^{G})$ and rewards $R_S(y_i \mid x) = \mathbb{I}[a(y_i) = \hat{a}(x)]$
    end if
    Estimate pseudo-correctness score $s_t(x) \leftarrow \frac{1}{G}\sum_{i=1}^{G} R_S(y_i \mid x)$
    Add failed tuples $(x, y_i, \hat{a}(x))$ with $R_S(y_i \mid x) = 0$ to $\mathcal{F}^{(t)}$
  end for
  Update Student policy π_{θ_t} by GRPO on trajectories collected from $\mathcal{D}_t$
  Reflect on a sampled subset of $\mathcal{F}^{(t)}$ to extract weakness descriptors $\mathcal{W}^{(t)}$
  Update memory:

$$\mathcal{M}_t \leftarrow \text{Evict}\Big(\text{Resolve}\big(\text{Merge}(\mathcal{M}_{t-1}, \mathcal{W}^{(t)}), \Delta\big), K\Big)$$

  Build synthesis prompt from sampled failures and persistent entries in $\mathcal{M}_t$
  Initialize next-round rollout cache $\mathcal{R}_{\text{var}}^{(t)} \leftarrow \emptyset$
  for $j = 1$ to M **do**
    Sample Teacher variant $x'_j \sim \pi_{\theta_t}(\cdot \mid \mathcal{F}^{(t)}, \mathcal{M}_t)$
    Pre-evaluate the variant with Student rollouts $\{y'_{j,i}\}_{i=1}^{G} \sim \pi_{\theta_t}(\cdot \mid h_t, x'_j)$
    Compute consensus answer $\hat{a}(x'_j)$, rewards $R_S(y'_{j,i} \mid x'_j)$, and pseudo-correctness $s_t(x'_j)$
    Compute frontier reward $R_{\text{diff}}(x'_j) = 4s_t(x'_j)(1 - s_t(x'_j))$
    Compute similarity penalty $R_{\text{sim}}(x'_j, x)$ against the source question and other variants
    Record Teacher reward $R_T(x'_j) = \max(0, R_{\text{diff}}(x'_j) - \lambda R_{\text{sim}}(x'_j, x))$
    Cache $\big(x'_j, \{y'_{j,i}\}_{i=1}^{G}, \hat{a}(x'_j), \{R_S(y'_{j,i} \mid x'_j)\}_{i=1}^{G}\big)$ into $\mathcal{R}_{\text{var}}^{(t)}$ for reuse in iteration $t+1$
  end for
  Set $\mathcal{X}_{\text{var}}^{(t)} \leftarrow \{x'_j\}_{j=1}^{M}$
end for
return Final adapted policy π_{θ_T}, weakness memory $\mathcal{M}_T$, rollout cache $\mathcal{R}_{\text{var}}^{(T)}$, and strategy note compiled from $\mathcal{M}_T$

C.1 Algorithmic Summary

Algorithm 1 summarizes the full TTSR procedure, including Student adaptation, Teacher reflection-guided synthesis, cross-iteration memory updates, and a rollout-reuse acceleration strategy that pre-evaluates synthesized variants and reuses these rollouts in the next iteration.

C.2 Similarity Metric Definition

To quantify the similarity between two questions in the similarity penalty, we adopt a sequence-based similarity measure that captures contiguous overlapping spans between two token sequences. This metric is designed to penalize near-duplicate generations while remaining robust to minor surface variations.

Sequence Representation. Given two questions represented as token sequences S_1 and S_2, we define their total length as

$$T = |S_1| + |S_2|, \tag{15}$$

where $|\cdot|$ denotes the number of tokens in the sequence.

Matching Blocks. We identify a set of non-overlapping matching blocks between S_1 and S_2. Each matching block is defined as a contiguous subsequence that appears in both sequences. The matching process first finds the longest contiguous matching subsequence and then recursively searches for additional matches in the remaining left and right segments. Let the resulting set of matching blocks be represented as

$$\mathcal{B} = \{(i, j, n)\},$$

where (i, j, n) denotes a matching block of length n starting at position i in S_1 and position j in S_2. The total number of matched tokens is then defined as

$$C = \sum_{(i,j,n)\in\mathcal{B}} n. \tag{16}$$

Similarity Ratio. Using the above definitions, we compute the normalized similarity ratio between S_1 and S_2 as

$$\mathrm{sim}(S_1, S_2) = \frac{2C}{T}, \tag{17}$$

which takes values in the range $[0, 1]$. A higher similarity ratio indicates greater overlap between the two sequences, while a lower value reflects more substantial differences in surface form. This similarity measure is used to compute the pairwise similarity scores required by the similarity penalty described in the main text (Section 3.2). By focusing on contiguous matching spans rather than isolated token overlap, the metric effectively discourages trivial paraphrasing while allowing moderate overlap that preserves shared reasoning structure.

C.3 Theoretical Analysis of the Difficulty Reward

The frontier-based difficulty reward $R_{\text{diff}}(x') = 4\, s_t(x')(1 - s_t(x'))$ draws on the insight—shared by recent zero-data self-play methods (Huang et al., 2025; Liu et al., 2025) and difficulty-filtered RL (Bae et al., 2025)—that training signal is maximized when the learner operates near its capability boundary. We formalize this principle through (A) an entropy-regularized RL analysis showing that the population analogue $p_t(1 - p_t)$ characterizes the *learning potential* of a question, and (B) a concentration bound establishing that the empirical estimate $s_t(x')$ converges uniformly to the population quantity.

Setup. For a fixed variant question x', each Student trajectory receives a binary pseudo-correctness reward $R_i \in \{0, 1\}$ (Eq. (7) in the main text), and the empirical score $s_t(x') = \frac{1}{G}\sum_{i=1}^{G} R_i$ (Eq. (8)) aggregates G rollouts. Let $p_t(x') = \mathbb{P}(R_i = 1 \mid x')$ denote the Student's underlying success probability at iteration t, so that $R_i \sim \mathrm{Bernoulli}\big(p_t(x')\big)$.

(A) Learning potential under entropy-regularized binary rewards. Consider the entropy-regularized optimal policy associated with the binary reward $R_S(\cdot \mid x')$:

$$\pi^*(y \mid x') \;\propto\; \pi_{\theta_t}(y \mid h_t, x') \,\exp\!\Big(\frac{R_S(y \mid x')}{\beta}\Big),$$

where $\beta > 0$ is the temperature parameter. In this setting, standard soft-RL analysis (Bae et al., 2025) shows that, for sufficiently large β, the reverse KL from the current Student policy to the soft-optimal policy admits the second-order expansion

$$D_{\mathrm{KL}}\big(\pi_{\theta_t}(\cdot \mid h_t, x') \,\big\|\, \pi^*(\cdot \mid x')\big) \;=\; \frac{p_t(x')\big(1 - p_t(x')\big)}{2\beta^2} \;+\; O\big(\beta^{-3}\big). \tag{18}$$

Therefore, up to higher-order terms, the policy gap induced by the binary pseudo-correctness reward is proportional to $p_t(x')(1-p_t(x'))$. This quantity is maximized at $p_t(x') = \frac{1}{2}$ and vanishes at the endpoints $p_t(x') \in \{0, 1\}$.

This has a natural interpretation in our setting. If $p_t(x') \approx 1$, the Student already solves the variant reliably, so the question is too easy to provide substantial additional learning signal. If $p_t(x') \approx 0$, the

Student almost never agrees with its own consensus signal, indicating that the question is likely too hard or too noisy under the current policy to provide reliable self-supervised feedback. In contrast, when $p_t(x') \approx 0.5$, the question lies near the Student's current capability frontier: it is neither trivial nor degenerate, and it retains the largest remaining learning potential under the pseudo-correctness objective.

Connection to the TTSR curriculum. We define the *population frontier score*

$$R^{\star}_{\text{diff}}(x') \;:=\; 4\,p_t(x')\big(1-p_t(x')\big) \;\in\; [0,1], \tag{19}$$

the normalized Bernoulli variance. Then Eq. (18) can be rewritten as $D_{\text{KL}}(\pi_{\theta_t}\|\pi^*) = R^{\star}_{\text{diff}}(x')/(8\beta^2) + O(\beta^{-3})$; that is, up to normalization and higher-order terms, $R^{\star}_{\text{diff}}(x')$ is proportional to a reverse-KL learning-potential proxy, which motivates frontier-oriented question selection in our curriculum design.

(B) Concentration of the empirical reward and uniform deviation bound. Because $s_t(x')$ is the empirical mean of G Bernoulli samples, Hoeffding's inequality gives

$$\mathbb{P}\big(|s_t(x') - p_t(x')| \geq \epsilon\big) \leq 2\exp(-2G\epsilon^2). \tag{20}$$

Let $f(u) = 4u(1-u)$. Since $|f'(u)| = |4(1-2u)| \leq 4$ for all $u \in [0,1]$, f is 4-Lipschitz, hence

$$|R_{\text{diff}}(x') - R^{\star}_{\text{diff}}(x')| = |f(s_t(x')) - f(p_t(x'))| \leq 4|s_t(x') - p_t(x')|.$$

Combining with Eq. (20), with probability at least $1-\alpha$,

$$|R_{\text{diff}}(x') - R^{\star}_{\text{diff}}(x')| \leq 4\sqrt{\frac{\log(2/\alpha)}{2G}}. \tag{21}$$

Therefore, $R_{\text{diff}}(x')$ is a uniformly consistent estimator of the population frontier score $R^{\star}_{\text{diff}}(x')$ as G increases.

Implication. Eq. (18) provides a principled justification linking the proposed difficulty reward to the learning potential under entropy-regularized RL: the Bernoulli variance $p_t(1-p_t)$ serves as a proxy for the remaining learning potential under the entropy-regularized pseudo-correctness objective. Eq. (21) further quantifies the statistical gap between the empirical reward $R_{\text{diff}}(x')$ and its population counterpart $R^{\star}_{\text{diff}}(x')$, ensuring that the curriculum selection remains faithful to the true frontier with controlled finite-sample error.

C.4 Prompt Templates

```
[ROLE: TEACHER]

Your task is to extract a generalizable reasoning weakness from an original question and a failed or unstable reasoning trace produced by a student model.
You must strictly base your analysis on the given inputs. Do not introduce external knowledge, do not assume access to the correct answer, and do not judge final correctness.
[ORIGINAL_QUESTION]
{text}

[FAILED_REASONING_TRACE]
{text}

[ANALYSIS GUIDELINES]
(1) Error Localization: Identify the first point where the reasoning becomes unreliable, incomplete, or invalid.
(2) Weakness Abstraction: Summarize the underlying reasoning weakness in one abstract sentence, without referring to specific values or variables.
(3) Trigger Conditions: Describe what problem structures or conditions are likely to trigger this weakness.
(4) Failure Signature: Describe typical reasoning patterns or behaviors when this weakness appears.
[OUTPUT REQUIREMENTS]
Output must be valid JSON and contain the following fields:
{
  "reasoning_weakness": "...",
  "trigger_conditions": ["..."],
  "failure_signature": ["..."],
  "localization_summary": "..."
}

[EXAMPLE]
[ORIGINAL_QUESTION]
Let x and y be real numbers such that x + y = 10 and x^2 + y^2 = 58. Find the maximum possible value of x.
[FAILED_REASONING_TRACE]
From x + y = 10, y = 10 - x. Substituting gives x^2 + (10 - x)^2 = 58. Solving yields x = 3 or x = 7. The reasoning directly selects 7 as the maximum.
[OUTPUT]
{
  "reasoning_weakness": "When multiple valid solutions exist, the reasoning directly selects
    the numerically larger solution without semantic justification.",
  "trigger_conditions": ["Multiple algebraic solutions", "Extremal objectives"],
   "failure_signature": ["Selecting the largest root immediately", "Equating magnitude with
optimality"],
  "localization_summary": "The reasoning deviates when it selects the larger algebraic solution
solely based on its value."
}
```

Table 4: Teacher prompt template used for reasoning weakness extraction in TTSR.

[ROLE: TEACHER]

Your task is to generate one new training question based on the original question, a failed reasoning trace, and an extracted reasoning weakness.
The goal is not to make the question arbitrarily harder, but to produce a question that is targeted, learnable, and close to the student's capability frontier.
You must ensure that:
(1) The new question preserves the core reasoning structure of the original question.
(2) The new question is more likely to trigger the given reasoning weakness.
(3) The new question is self-contained, solvable, and unambiguous.
(4) The new question is not a superficial paraphrase or simple numerical substitution.

[INPUT]
[ORIGINAL_QUESTION]
{text}

[FAILED_REASONING_TRACE]
{text}

[WEAKNESS_JSON]
{reasoning_weakness, trigger_conditions, failure_signature}

[STEP-BY-STEP INSTRUCTIONS]
Step 1: Anchor Structure.
Summarize the core reasoning structure of the original question, including key concepts, variable relationships, constraints, and the main reasoning steps required for a correct solution.
Use structural language rather than restating the original question.

Step 2: Error-Hitting Strategy.
Explicitly state how the new question will be designed to more reliably trigger the given reasoning weakness, while remaining solvable and fair.
Explain what aspects will be modified, what shortcuts are targeted, and how superficial paraphrasing is avoided.
Step 3: Generate One New Question.
Generate a single, fully self-contained question that requires multi-step reasoning and is likely to trigger the weakness under common shortcuts.
Avoid changing only numbers, swapping story contexts, or adding irrelevant complexity.

Step 4: Hit Rationale.
Briefly explain why the generated question is likely to expose the reasoning weakness and how a correct solution would avoid the shortcut.
Step 5: Self-Test Filter.
Verify whether the question is likely to trigger the weakness, remains learnable, and is not a surface-level paraphrase.
If any check fails, regenerate the question.

Table 5: Teacher prompt template used for reflection-guided question synthesis in TTSR.

```
[OUTPUT REQUIREMENTS]
The output must be valid JSON. Do not include markdown or additional explanations.
All fields must be present and filled according to the schema below.

{
  "anchor_structure": [
    "...core structural element 1...",
    "...core structural element 2..."
  ],
  "error_hitting_strategy": {
    "what_to_avoid": ["...simple variants that would not trigger the weakness..."],
    "what_to_add": ["...structural modifications to increase weakness exposure..."],
    "shortcut_to_block": ["...erroneous shortcut to be induced or blocked..."],
    "fairness_check": "...how solvability and unambiguity are ensured..."
  },
  "generated_question": "...fully self-contained question text...",
  "hit_rationale": [
    "...why the question is likely to trigger the weakness...",
    "...what a correct reasoning process would need to check..."
  ],
  "self_test": {
    "likely_to_trigger_weakness": "YES/NO + brief reason",
    "learnable_frontier": "YES/NO + brief reason",
    "not_surface_paraphrase": "YES/NO + brief reason"
  }
}
```

Table 6: Teacher prompt template used for reflection-guided question synthesis in TTSR (continued).

D Case Study

To qualitatively illustrate how the TTSR pipeline operates, we present two representative case studies from Qwen3-8B-Base trained on MATH500.

Full pipeline trace (Table 7). Table 7 traces a single training instance through the complete TTSR pipeline. The Student encounters a quartic equation and solves it via the standard substitution $t = x^2$, but recovers only the positive roots—a failure pattern commonly observed in early iterations. The Teacher analyzes the failed trace and extracts a structured weakness descriptor pinpointing the incomplete back-substitution. Based on this weakness, the Teacher synthesizes a variant question that algebraically reduces to the same equation but asks for the *sum* of all solutions, making the back-substitution error directly observable in the final answer.

Strategy note effectiveness (Table 8). Table 8 demonstrates how the accumulated weakness memory translates into actionable reasoning guidance. After the back-substitution weakness is recorded and persists across multiple iterations, it is compiled into a strategy note prepended to the Student's prompt. When the Student encounters a new even-power equation at a later iteration, the strategy note explicitly reminds it to check for negative branches—resulting in a correct solution that the earlier Student would have missed. This before-and-after comparison highlights how TTSR's memory mechanism enables *targeted, persistent* reasoning improvement, rather than relying solely on gradient updates.

Variant evolution across iterations (Table 9). Table 9 tracks how the Teacher's synthesized variant questions evolve over training iterations for two representative MATH500 problems. In early iterations (Iter 2), variants are surface-level modifications—changing coefficients or base/modulus while preserving the same solution technique. As the weakness memory accumulates, the Teacher produces increasingly targeted variants: at Iter 5, variants *amplify* the identified weakness (e.g., requiring Newton's identity chains instead of single-step Vieta's); at Iter 8, variants *transfer* the weakness to a different mathematical domain (e.g., reciprocal-squared roots, or modular arithmetic with composite moduli); by Iter 10, variants *integrate* multiple weaknesses into a single problem requiring simultaneous application of several reasoning techniques. This progression demonstrates that TTSR's reflection-guided synthesis produces a natural curriculum—from surface variation to deep structural targeting—without any explicit difficulty scheduling.

E Variant Quality Analysis

E.1 Frontier Score Distribution

Table 10 reports the pseudo-correctness score distribution of three question sources on Qwen3-8B-Base (AIME25): original test questions, variants synthesized without reflection (random structural variation), and TTSR variants with full reflection-guided synthesis.

TTSR variants exhibit a substantially higher frontier ratio (64.2%) than both original questions (8.3%) and reflection-free variants (35.6%), confirming that weakness-aware diagnosis effectively concentrates synthesized questions near the Student's capability boundary.

E.2 Variant Structural Quality (GPT-5.2 Evaluation)

Beyond frontier alignment, a key design goal is that TTSR variants target the *specific weakness* identified by the Teacher's reflection, rather than being superficial modifications (e.g., changing numbers or swapping variable names). To evaluate this, we use GPT-5.2 (2025-12-11) as an external judge on $n = 100$ MATH500 and $n = 50$ AIME25 variant instances. Each instance is scored along two binary dimensions:

- **Weakness Targeting (WT)**: Does the variant specifically test the reasoning weakness identified in the Teacher's reflection descriptor?
- **Structural Novelty (SN)**: Does the variant involve meaningful structural changes (e.g., domain transfer, constraint modification, multi-step integration) beyond simple number substitution or surface paraphrasing?

Without reflection, the majority of variants (59% on MATH500, 66% on AIME25) are trivial modifications—changing coefficients, swapping story contexts, or adding irrelevant complexity. TTSR's reflection-guided synthesis raises the structural novelty rate to 78% and 72%, while additionally targeting the identified weakness in 83% and 76% of cases. The joint rate (WT$\wedge$SN) of 71%

[Original Question] *(Source: MATH500, Iteration 3)*
Find the number of distinct real solutions to the equation $x^4 - 13x^2 + 36 = 0$.

[Student's Failed Reasoning Trace]
Let $t = x^2$. Then $t^2 - 13t + 36 = 0$.
Using the quadratic formula: $t = \frac{13 \pm \sqrt{169-144}}{2} = \frac{13 \pm 5}{2}$.
So $t = 9$ or $t = 4$.
From $t = 9$: $x = 3$. From $t = 4$: $x = 2$.
Therefore, there are **2** distinct real solutions. (× Correct answer: **4**)

[Teacher's Extracted Weakness]

```
{
  "reasoning_weakness": "After solving via even-power substitution,
    only the principal (positive) root is recovered, omitting the negative branch.",
  "trigger_conditions": ["Even-power variable substitution (t = x^n, n even)",
      "Requests for the count or aggregate of all real solutions"],
  "failure_signature": ["Only non-negative roots reported",
      "Solution count is exactly half the true count"],
  "localization_summary": "Deviation occurs at back-substitution:
    t = 9 yields x = 3 only, instead of x = ±3."
}
```

[Teacher's Synthesized Variant Question]
Let S be the set of all real numbers x satisfying $(x^2 - 4)(x^2 - 9) = 0$. Compute the *sum* of the elements of S.
Hit rationale: Expanding yields $x^4 - 13x^2 + 36 = 0$—the same substitution equation. However, asking for the **sum** (rather than the count) makes the error observable: the incorrect answer is $2 + 3 = 5$, while the correct answer accounting for $\pm$ branches is $(-3) + (-2) + 2 + 3 = 0$.

Table 7: **Case study: full TTSR pipeline on a representative example.** We trace one training instance through the complete pipeline—from the original question through the Student's failed reasoning, the Teacher's weakness extraction, and the synthesized variant question. Color coding: original question, failed trace, Teacher reflection, variant question.

and 64% indicates that the majority of TTSR variants are both structurally non-trivial *and* weakness-targeted. AIME25 scores are systematically lower, consistent with the reflection quality evaluation in Appendix M: competition-level problems produce longer reasoning chains where weakness localization is harder.

E.3 Pseudo-Label Accuracy (GPT-5.2 Verification)

We track the majority-vote accuracy of the Student's training batch across iterations on AIME25 (Qwen3-8B-Base). For original test questions, correctness is determined by the ground-truth answer; for synthesized variants (which have no canonical answer), we use GPT-5.2 as an oracle judge. Figure 5 reports the results.

Both methods start near 13% at iteration 2, reflecting the extreme difficulty of AIME25 (base accuracy 9.8%). TTRL, which trains only on the original questions, improves slowly and with fluctuation, reaching 17.2% at convergence. TTSR introduces easier, targeted variant questions that provide more learnable samples and simultaneously raise the overall majority-vote accuracy to 25.4% (+8.2 over TTRL). The gap widens primarily in later iterations as the weakness memory accumulates and the Teacher synthesizes increasingly well-calibrated variants.

F Out-of-Distribution Performance

To assess the transferability of test-time training across different mathematical benchmarks, we train on each of the six source benchmarks independently and evaluate on all six benchmarks. Figure 6 reports the results on Qwen3-8B-Base.

Cross-benchmark transfer. TTSR outperforms TTRL in the majority of OOD evaluations, with gains typically ranging from 1–3 points. For instance, when trained on Olympiad and evaluated on AIME24, TTSR reaches 25.4 compared to TTRL's 22.0 (+3.4), whereas the Base model achieves only 15.9. Notably, a small number of OOD transfer pairs exhibit comparable or slightly reversed results (e.g., Olympiad→MATH500: TTRL 81.2 vs. TTSR 80.8), which is expected given the stochastic

[New Question] *(Encountered at Iteration 7)*
Find the sum of all real values of x satisfying $x^6 = 64$.

[Student Reasoning — Without Strategy Note] *(Iteration 3 behavior)*
$x^6 = 64$. Taking the sixth root of both sides: $x = 64^{1/6} = 2$.
The sum of all real values is **2**. ($\times$ Correct answer: **0**)

[Compiled Strategy Note] *(From weakness memory at Iteration 7)*
[REASONING STRATEGY NOTES]
Based on recurring reasoning patterns, keep the following in mind:
1. When solving equations via substitution $t = x^n$ with n even, always recover **both** $\pm\sqrt[n]{t}$ during back-substitution. Verify that the solution count matches the degree of the original equation.
2. When computing aggregates (sum, product) over all solutions, enumerate every branch before combining.

[Student Reasoning — With Strategy Note] *(Iteration 7 behavior)*
$x^6 = 64$. Since the exponent 6 is even, x can be positive or negative.
$|x|^6 = 64 \implies |x| = 2$, so $x = 2$ or $x = -2$.
Verification: $2^6 = 64$ ✓; $(-2)^6 = 64$ ✓.
The sum of all real values is $2 + (-2) =$ **0**. (✓ Correct)

Table 8: **Case study: strategy note effectiveness on a subsequent problem.** After the weakness from Table 7 is accumulated in the weakness memory, the compiled strategy note guides the Student to avoid the same error on a new problem. Color coding: reasoning without note, strategy note, reasoning with note.

nature of cross-benchmark generalization and the relatively small OOD gaps.

Role of source difficulty. Training on harder benchmarks (AIME24, AIME25, Olympiad) generally produces stronger OOD transfer than training on easier ones (AMC23, MATH500). For example, models trained on AIME24 achieve higher OOD scores on Olympiad (49.0 vs. 44.6 from AMC23 training), suggesting that harder source problems induce more generalizable reasoning refinements. TTSR amplifies this effect through reflection-guided synthesis: by targeting the model's specific weaknesses, the generated variant questions capture transferable reasoning patterns rather than benchmark-specific shortcuts.

G Additional Training Dynamics

To complement the MATH500-based training dynamics presented in Figure 4 (main text), we report additional training dynamics when TTSR is applied with AMC23 and Olympiad as primary training benchmarks.

AMC23 (top row). On the in-domain AMC23 benchmark, TTSR trails TTRL at iteration 2 but overtakes it by iteration 4 and reaches 66.5 at convergence, a +5.0 margin over TTRL (61.5). A mid-training dip at iteration 6 mirrors the pattern observed on MATH500 in the main text. Cross-math transfer to AIME25 is moderate (17.6), reflecting the distributional gap between competition-format AMC problems and the AIME format.

Olympiad (bottom row). Training on Olympiad produces a qualitatively different trajectory: TTSR jumps ahead by iteration 4 (51.3 vs. 47.4) and, after a brief dip at iteration 6, resumes climbing to reach 55.2 (+4.4 over TTRL) at convergence. TTRL exhibits a mid-training dip at iteration 8 (49.8 → 48.6) before recovering, whereas TTSR's gap over TTRL widens steadily from iteration 6 onward. Cross-math transfer to AIME25 reaches 17.8, and both rows confirm that cross-domain gains on GPQA-D and MMLU-Pro remain positive but modest.

H Self-Consistency Comparison

To disentangle the effect of inference-time scaling from test-time training, we compare against a **Self-Consistency** (SC) (Wang et al., 2022) baseline that applies majority voting over $G = 16$ sampled reasoning trajectories per question *without any parameter updates*. Table 12 reports results for all three base models across eight benchmarks.

Self-Consistency provides moderate gains over greedy decoding (Base Model) by leveraging sampling diversity: +4.2 on Qwen3-4B, +3.4 on Qwen3-8B, and +1.8 on OctoThinker-8B. However, all test-time training methods substantially outperform SC. Even R-Zero, the weakest TTT baseline, surpasses SC on average, and TTSR's

Iteration	Case 1: Symmetric Root Functions	Case 2: Modular Exponentiation
Test Question	Let r_1, r_2, r_3 be the roots of $x^3 - 7x + 6 = 0$. Find $r_1^2 + r_2^2 + r_3^2$.	Find the remainder when 2^{100} is divided by 7.
ExtractedWeakness	*"Applies single-step Vieta's formula correctly but fails to chain identities for higher-order or non-standard symmetric functions of roots."*	*"Identifies cyclic patterns in modular arithmetic but fails to decompose nested exponents or composite moduli requiring multi-level analysis."*
Iter 2	Find $r_1^2+r_2^2+r_3^2$ for the roots of $x^3-3x^2+x-5=0$. [Surface: changed coefficients, same one-step Vieta]	Find the remainder when 3^{200} is divided by 13. [Surface: different base/modulus, same cycle technique]
Iter 5	For the roots r_i of $x^3 - 4x^2 + 5x - 2 = 0$, compute $r_1^3 + r_2^3 + r_3^3$. [Weakness amplification: cubic power sum requires Newton's identity chain, not one-step Vieta]	What is the units digit of 7^{7^7}? [Weakness amplification: nested exponent requires two-level cycle decomposition]
Iter 8	If the roots of $x^3 - 6x^2 + 11x - 6 = 0$ are r_i, find $\frac{1}{r_1^2} + \frac{1}{r_2^2} + \frac{1}{r_3^2}$. [Domain transfer: reciprocal-squared form shifts from polynomial to rational algebra]	Find the last two digits of 3^{2025}. [Domain transfer: mod 100 requires Euler's theorem or CRT decomposition]
Iter 10	The roots of $x^3 - x^2 - 4x + 4 = 0$ are a, b, c. Find $\sum_{\text{cyc}} \frac{a^2}{b+c}$. [Multi-weakness: fractional symmetric function combines identity chaining with constraint $b+c = S_1 - a$]	Determine the remainder when $2^{2^{2^3}}$ is divided by 1000. [Multi-weakness: tower exponent + 3-prime modulus combines nested cycle with CRT]

Table 9: **Evolution of reflection-guided variant questions across training iterations.** Two representative cases from MATH500 are shown. Unlike generic question synthesis, each variant is guided by the Teacher's *extracted weakness*, resulting in progressively more targeted and structurally diverse problems. Blue: surface variation; Red: weakness amplification; Green: domain transfer; Purple: multi-weakness integration.

Question Source	Mean s_t	Std	Frontier %
Original questions	0.12	0.21	8.3
w/o Reflection variants	0.31	0.26	35.6
TTSR variants	0.47	0.19	64.2

Table 10: Variant quality on Qwen3-8B-Base (AIME25). Frontier ratio: fraction of questions with $s_t \in [0.2, 0.8]$.

margin over SC is +7.5, +8.6, and +13.6 for the three models respectively. These results confirm that TTSR's improvements stem from genuine parameter adaptation guided by reflection and variant synthesis, rather than from the additional inference-time computation inherent in sampling multiple trajectories.

I Full Component-Level Ablation

Table 13 complements the additive analysis in Table 2 (main text) by removing each component individually from the full TTSR system on Qwen3-8B-Base.

Averaging absolute drops across benchmarks, the individual components rank as: Teacher co-evolution (-5.3) $>$ reflection-guided synthesis (-4.2) $>$ difficulty reward (-3.0) $>$ weakness memory (-2.5) $>$ similarity penalty (-2.2) $>$ strategy note (-1.4). The two largest contributors—co-evolution and reflection—jointly define TTSR's core curriculum mechanism, while the remaining components provide complementary but individually smaller gains.

Compound ablation: Reflection + Strategy Note Only. Table 14 isolates the contribution of the auxiliary reward-shaping and memory components by retaining only the reflection mechanism and strategy note (i.e., removing difficulty reward, similarity penalty, and weakness memory simultaneously).

Reflection and strategy note alone recover a majority of TTSR's total gain on easier benchmarks (e.g., 69% on MATH500, 63% on Olympiad) but a smaller fraction on harder ones (43% on AIME25), suggesting that the auxiliary components—especially difficulty reward and weakness memory—become increasingly important as task difficulty rises. The remaining +3.0–

Method	Benchmark	WT (%)	SN (%)	WT∧SN (%)
w/o Reflection	MATH500	—	41.0	—
w/o Reflection	AIME25	—	34.0	—
TTSR	MATH500	83.0	78.0	71.0
TTSR	AIME25	76.0	72.0	64.0

Table 11: GPT-5.2 (2025-12-11) structural evaluation of synthesized variants on Qwen3-8B-Base. WT: weakness targeting rate; SN: structural novelty rate; WT∧SN: both criteria satisfied jointly.

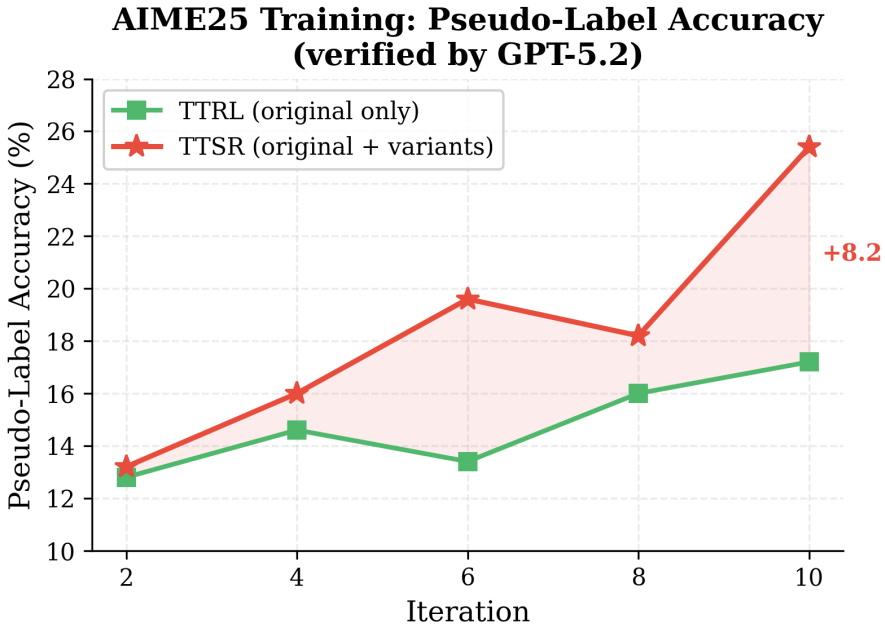

Figure 5: Majority-vote accuracy during AIME25 training (Qwen3-8B-Base). Original questions are verified by ground truth; variant questions are verified by GPT-5.2 (2025-12-11).

+5.3 points contributed by difficulty reward, similarity penalty, and weakness memory confirm that these components substantially amplify the curriculum's effectiveness by shaping the reward landscape and enabling cross-iteration adaptation.

J Computational Cost Analysis

We analyze the per-iteration computational overhead of TTSR relative to TTRL on Qwen3-4B-Base under the same runtime setup as the main experiments ($M=2$, $G=16$, $T=10$).

Per-iteration breakdown. Table 15 reports cost using a unified normalization: one TTRL iteration is set to $1.0\times$. The dominant overhead in TTSR comes from (i) Student rollouts on synthesized variants and (ii) larger GRPO updates over the mixed batch. Teacher reflection/synthesis are single-pass generations and contribute relatively little overhead. Overall, one TTSR iteration is about $3.0\times$ one TTRL iteration.

Iso-compute comparison. Since each TTSR iteration costs $\sim 3\times$ that of TTRL, running TTSR for $T=10$ iterations consumes roughly the same budget as TTRL for 30 iterations. Table 16 reports this comparison on all eight benchmarks in the main text (Qwen3-4B-Base).

Extending TTRL from 10 to 30 iterations yields only $+0.9$ additional points ($42.7 \rightarrow 43.6$), indicating clear saturation. Under the same compute budget, TTSR achieves 47.5 ($+3.9$ over extended TTRL), confirming that the gains stem from the reflection-guided curriculum rather than additional compute. The training dynamics in Figure 4 corroborate this: TTRL's accuracy oscillates in later iterations without sustained improvement, whereas TTSR continues to make progress through targeted variant synthesis.

K Statistical Robustness

To verify that TTSR's improvements are not artifacts of a particular random seed, we run both

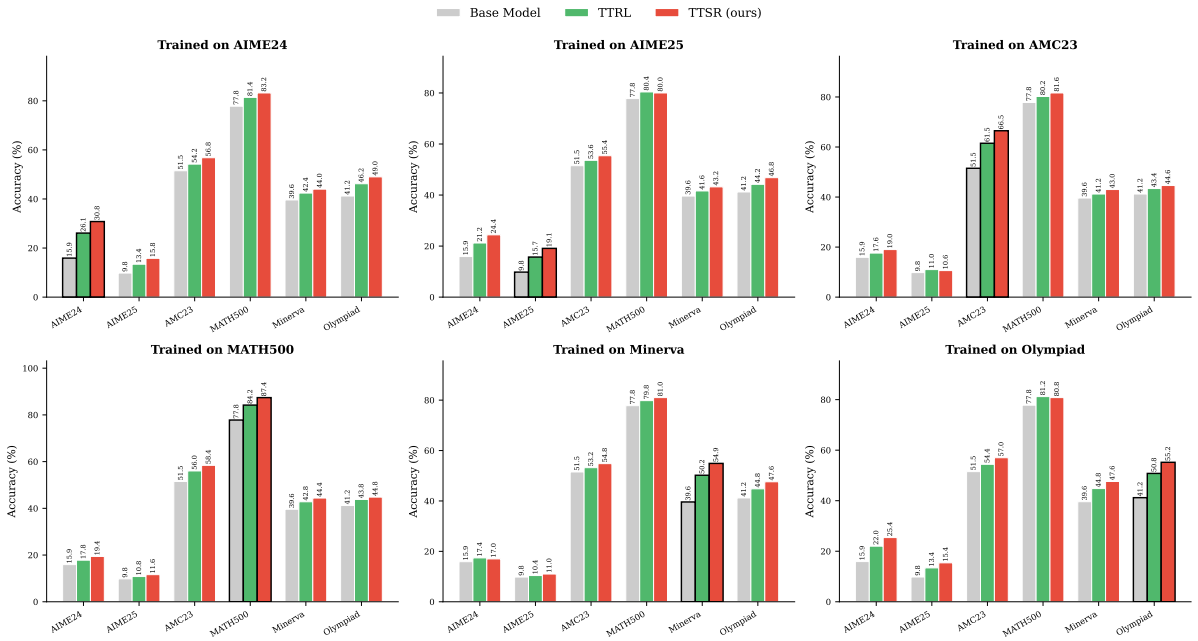

Figure 6: **Out-of-distribution performance comparison on Qwen3-8B-Base.** Each subplot shows the accuracy when the model is trained on a single source benchmark (title) and evaluated on all six mathematical reasoning benchmarks. In-domain bars (where train=eval) are highlighted with black borders. TTSR generally outperforms TTRL across most OOD evaluations, though a few individual transfer pairs show comparable or slightly reversed performance, reflecting natural variance in cross-benchmark generalization.

TTSR and TTRL with three independent seeds on Qwen3-4B-Base. Table 17 reports mean and standard deviation across seeds on four representative benchmarks.

TTSR outperforms TTRL on three of the four benchmarks under the reported seed averages, with non-overlapping confidence intervals on MATH500, AIME25, and GPQA-D, while Olympiad remains slightly below TTRL. TTSR also exhibits lower variance across seeds (avg std 0.68 vs TTRL's 1.03), suggesting that the reflection-guided curriculum provides more stable training dynamics than TTRL's direct adaptation on raw test questions. For Qwen3-8B-Base and OctoThinker-8B, we report single-seed results; however, the consistent trends across three model backbones and eight benchmarks in Table 1 provide complementary evidence of robustness.

L Memory Evolution Analysis

To understand how the weakness memory $\mathcal{M}_t$ evolves during test-time training, we track its composition across iterations on Qwen3-4B-Base (MATH500, $K = 10$, $\Delta = 3$).

Memory dynamics. Table 18 reports the memory size, newly added entries, resolved (evicted) entries, and persistent entries (surviving ≥ 3 iterations) at each iteration.

The memory exhibits three phases. In iterations 1–2, new weaknesses are discovered rapidly as the model encounters diverse failure modes, and the memory fills to its capacity $K = 10$ by iteration 2 (*discovery phase*). In iterations 3–4, the memory remains at capacity as new entries roughly balance resolved ones (*maintenance phase*). From iteration 5 onward, fewer new weaknesses emerge while existing ones continue to be resolved, causing the memory to shrink steadily to 4 by iteration 10 (*convergence phase*). This pattern is consistent with the ablation finding that removing the weakness memory degrades performance by 1.8–3.2 points: the memory enables the Teacher to prioritize persistent weaknesses that the Student has not yet addressed.

Weakness category shift. We manually categorize the weakness descriptors into five types: algebraic manipulation, logical deduction, domain transfer, edge-case handling, and multi-step planning. In early iterations, algebraic and logical weaknesses dominate ($\sim 65\%$). As these are progressively resolved, domain transfer and edge-case weaknesses gain prominence at mid-training, and multi-step planning weaknesses become the largest

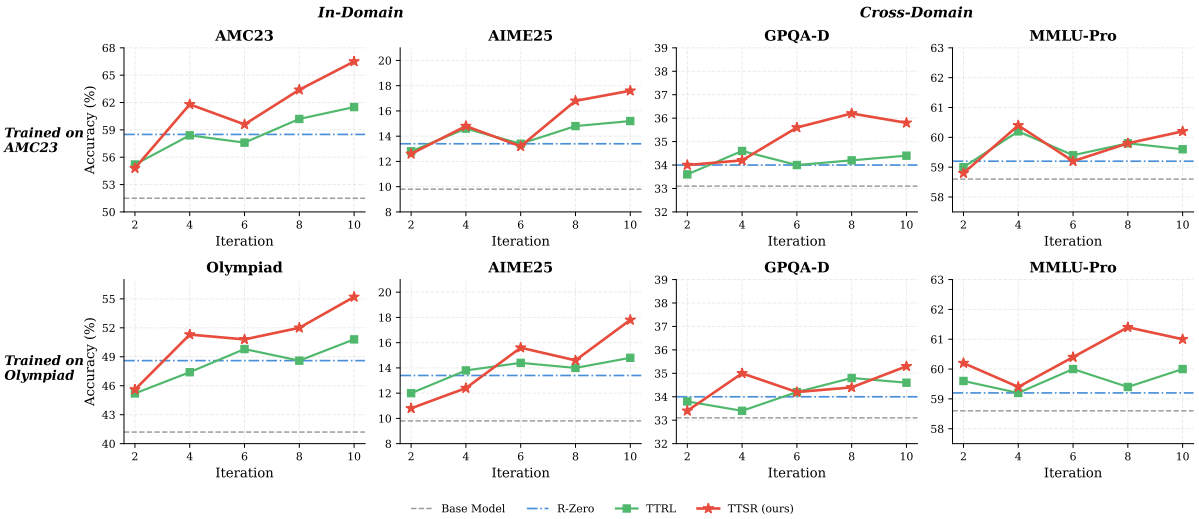

Figure 7: **Training dynamics on additional mathematical benchmarks (Qwen3-8B-Base).** Each row corresponds to a separate experiment where TTSR is applied with the indicated primary training benchmark. Left two panels show *in-domain* accuracy (primary benchmark and AIME25); right two panels show *cross-domain* accuracy on held-out general reasoning benchmarks (GPQA-D, MMLU-Pro). Dashed and dash-dotted horizontal lines denote the Base Model and R-Zero baselines, respectively.

category by iteration 10 ($\sim 31\%$). This natural progression from surface-level to structural weaknesses mirrors the variant evolution observed in the case studies (Table 9).

M Reflection Quality Evaluation

To assess whether the Teacher's reflections accurately identify genuine reasoning weaknesses, we conduct an automated evaluation using GPT-5.2 as an external judge on Qwen3-4B-Base.

Setup. We randomly sample 100 reflection instances from MATH500 and 30 from AIME25. Each instance consists of the original question, the Student's failed reasoning trace, and the Teacher's extracted weakness descriptor (including weakness summary, trigger conditions, and failure signature). GPT-5.2 evaluates each instance along three dimensions:

- **Weakness Accuracy**: Does the extracted weakness correctly identify the actual reasoning error in the Student's trace?
- **Actionability**: Is the weakness descriptor specific enough to guide targeted question synthesis?
- **Variant Alignment**: Does the subsequently synthesized variant question actually target the identified weakness?

Each dimension is scored as *yes* or *no*. We report the fraction of positive judgments.

On MATH500, the Teacher correctly identifies the reasoning weakness in 82% of cases, and the subsequent variant targets this weakness 79% of the time. Scores are lower on AIME25, which is expected given the higher difficulty: competition-level problems involve longer reasoning chains where pinpointing the first error is harder. The actionability scores (88% and 83%) are consistently higher than accuracy, indicating that even when the weakness diagnosis is imprecise, the descriptor typically provides enough structural information to guide useful variant synthesis. These results support the ablation finding that reflection-guided synthesis outperforms untargeted variant generation.

Method	AMC23	MATH500	Minerva	Olympiad	AIME24	AIME25	GPQA-D	MMLU-Pro	Δ
Qwen3-4B-Base									
Base Model	45.5	72.0	32.4	40.2	12.4	5.8	25.7	52.0	–
SC	53.5	77.0	37.9	43.2	16.8	8.8	28.2	54.5	+4.2
R-Zero	54.0	76.8	40.7	44.0	18.2	9.1	28.4	55.0	+5.0
TTRL	55.5	79.0	43.6	46.0	17.6	14.7	29.1	56.0	+6.9
TTSR (ours)	**61.0**	**82.4**	**53.0**	**45.3**	**25.6**	**17.6**	**34.2**	**60.8**	**+11.7**
Qwen3-8B-Base									
Base Model	51.5	77.8	39.6	41.2	15.9	9.8	33.1	58.6	–
SC	57.5	81.8	44.1	43.7	19.9	12.3	35.1	60.6	+3.4
R-Zero	58.5	82.0	47.8	48.6	22.6	13.4	36.7	61.5	+5.5
TTRL	61.5	84.2	50.2	50.8	26.1	15.7	38.4	62.8	+7.8
TTSR (ours)	**66.5**	**87.4**	**54.9**	**55.2**	**30.8**	**19.1**	**42.6**	**66.7**	**+12.0**
OctoThinker-8B-Hybrid-Base									
Base Model	28.5	45.6	15.2	16.8	7.1	3.9	15.2	26.8	–
SC	31.5	48.6	17.2	18.3	8.6	4.9	16.2	28.3	+1.8
R-Zero	35.5	54.0	22.9	25.4	11.6	6.8	19.7	31.4	+6.0
TTRL	38.5	57.2	26.4	28.7	13.9	8.2	21.5	33.2	+8.6
TTSR (ours)	**46.5**	**64.8**	**34.1**	**36.8**	**19.7**	**12.4**	**27.9**	**39.8**	**+15.4**

Table 12: Comparison with Self-Consistency (SC) baseline. SC applies majority voting over $G = 16$ sampled trajectories without parameter updates.

Settings	MATH500	AIME25	Olympiad	GPQA-D
TTSR (Full)	87.4 —	19.1 —	55.2 —	42.6 —
w/o Reflection-Guided Synthesis	85.0 ↓2.4	14.2 ↓4.9	49.8 ↓5.4	38.4 ↓4.2
w/o Teacher Test-Time Update	82.8 ↓4.6	13.3 ↓5.8	49.4 ↓5.8	37.8 ↓4.8
w/o Difficulty Reward	85.2 ↓2.2	15.4 ↓3.7	51.8 ↓3.4	39.8 ↓2.8
w/o Similarity Penalty	86.0 ↓1.4	16.4 ↓2.7	52.8 ↓2.4	40.2 ↓2.4
w/o Weakness Memory	85.6 ↓1.8	15.9 ↓3.2	52.4 ↓2.8	40.4 ↓2.2
w/o Strategy Note	86.4 ↓1.0	17.2 ↓1.9	53.6 ↓1.6	41.4 ↓1.2

Table 13: Full component-level ablation on Qwen3-8B-Base. Each row removes one component from the full TTSR system. Red arrows indicate absolute drops from the full model.

Configuration	MATH500	AIME25	Olympiad	GPQA-D
Base Model	77.8	9.8	41.2	33.1
Reflection + Strategy Note Only	84.4	13.8	50.0	38.2
TTSR (Full)	**87.4**	**19.1**	**55.2**	**42.6**

Table 14: Compound ablation on Qwen3-8B-Base: retaining only reflection-guided synthesis and strategy note while removing difficulty reward, similarity penalty, and weakness memory.

Component	TTRL	TTSR (relative)
Base TTRL iteration	1.0×	1.0×
Extra Student rollouts on variants ($M = 2$)	—	+1.0×
Extra GRPO on mixed batch ($(1 + M) = 3$)	—	+0.9×
Teacher Reflection	—	+0.05×
Teacher Synthesis	—	+0.08×
Memory & Note Update	—	negligible
Total per iteration	**1.0×**	**∼3.0×**

Table 15: Per-iteration computational breakdown on Qwen3-4B-Base. Normalization: one TTRL iteration = 1.0×.

Method	Iter.	Rel. Cost	AMC23	MATH500	Minerva	Olympiad	AIME24	AIME25	GPQA-D	MMLU-Pro	Avg
Base Model	—	$1\times$	45.5	72.0	32.4	40.2	12.4	5.8	25.7	52.0	35.8
TTRL	10	$10\times$	55.5	79.0	43.6	46.0	17.6	14.7	29.1	56.0	42.7
TTRL (extended)	30	$30\times$	57.1	81.0	44.3	46.4	17.8	14.8	30.0	57.4	43.6
TTSR	10	$\sim 30\times$	**61.0**	**82.4**	**53.0**	45.3	**25.6**	**17.6**	**34.2**	**60.8**	**47.5**

Table 16: Iso-compute comparison on Qwen3-4B-Base across all eight benchmarks in the main text. TTRL (extended) runs for 30 iterations to approximately match TTSR's total compute; its per-benchmark values are estimated from the observed test-time trajectory under the same setting.

Method	MATH500	AIME25	Olympiad	GPQA-D
TTRL	79.0 ± 0.8	14.7 ± 1.3	46.0 ± 1.1	29.1 ± 0.9
TTSR	$\mathbf{82.4 \pm 0.5}$	$\mathbf{17.6 \pm 0.9}$	$\mathbf{45.3 \pm 0.7}$	$\mathbf{34.2 \pm 0.6}$

Table 17: Multi-seed results on Qwen3-4B-Base (3 seeds). Mean $\pm$ std reported.

Iter	$\lvert\mathcal{M}_t\rvert$	New	Resolved	Persistent
1	6	6	0	0
2	10	5	1	0
3	10	4	4	3
4	10	3	3	5
5	9	2	3	5
6	8	2	3	4
7	7	1	2	4
8	6	1	2	3
9	5	1	2	3
10	4	0	1	2

Table 18: Weakness memory dynamics on Qwen3-4B-Base (MATH500, $K = 10$).

Benchmark	Accuracy	Actionability	Alignment
MATH500 ($n = 100$)	82.0%	88.0%	79.0%
AIME25 ($n = 30$)	76.7%	83.3%	73.3%

Table 19: GPT-5.2 (2025-12-11) evaluation of Teacher reflection quality on Qwen3-4B-Base.