Sticker-TTS: Learn to Utilize Historical Experience with a Sticker-driven Test-Time Scaling Framework

Anonymous ACL submission

Abstract

Large reasoning models (LRMs) have exhibited strong performance on complex reasoning tasks, with further gains achievable through increased computational budgets at inference. However, current test-time scaling methods predominantly rely on redundant sampling, ignoring the historical experience utilization, thereby limiting computational efficiency. To overcome this limitation, we propose **Sticker-TTS**, a novel test-time scaling framework that coordinates three collaborative LRMs to iteratively explore and refine solutions guided by historical attempts. At the core of our framework are distilled key conditions—termed *stickers*—which drive the extraction, refinement, and reuse of critical information across multiple rounds of reasoning. To further enhance the efficiency and performance of our framework, we introduce a two-stage optimization strategy that combines imitation learning with self-improvement, enabling progressive refinement. Extensive evaluations on three challenging mathematical reasoning benchmarks, including AIME-24, AIME-25, and OlymMATH, demonstrate that Sticker-TTS consistently surpasses strong baselines, including self-consistency and advanced reinforcement learning approaches, under comparable inference budgets. These results highlight the effectiveness of sticker-guided historical experience utilization.

1 Introduction

Recent advancements in foundation models, particularly when combined with reinforcement learning (RL) during training, have significantly improved the capabilities of LRMs on complex inference tasks (Team et al., 2025; Guo et al., 2025; Yang et al., 2025; Zhao et al., 2023). Empirical studies demonstrate that increasing computational budgets during both training and inference phases yields consistent gains in reasoning performance. For example, OpenAI's reasoning series models (*e.g.,* o1 and o3) highlight how test-time scaling can further boost accuracy on challenging benchmarks (OpenAI, 2024a,b, 2025). Unlike RL-based optimization—which incurs substantial computational overhead—test-time scaling offers a more affordable alternative, attracting growing interest for its favorable cost–performance trade-off (Chen et al., 2024; Kang et al., 2024).

Existing researches mainly propose two lines of approaches for achieving test-time scaling. A common approach executes multiple independent single-round inferences and selects the final answer via majority vote (Wang et al., 2022). Despite its simplicity and robustness as a strong baseline (Jiang et al., 2024), this strategy treats each inference as isolated, often resulting in redundant or uninformative computations. To address this limitation, recent studies have proposed an iterative multi-round inference method, where the model incorporates prior reasoning traces or final answers into subsequent inference inputs (Chen et al., 2025). While this paradigm encourages history-aware reasoning, it introduces new challenges: overly verbose reasoning histories in the input may lead models to forget or overlook salient facts, and the brevity of final answers makes it difficult for models to revise earlier outputs, even when faced with inconsistencies or superior alternatives. These issues become increasingly pronounced as reasoning chains grow in length and complexity.

To address the aforementioned challenges, we propose a novel framework aimed at striking a balance between overly verbose reasoning traces and excessively concise final answers, thereby encouraging LRMs to explore novel solution paths by leveraging historical attempts. Inspired by how humans approach long-form generative tasks—such as writing—by distilling key intermediate ideas, conclusions, or inflection points to scaffold the final output, we introduce a method to distill a compact set of essential solution cues from the lengthy rea-

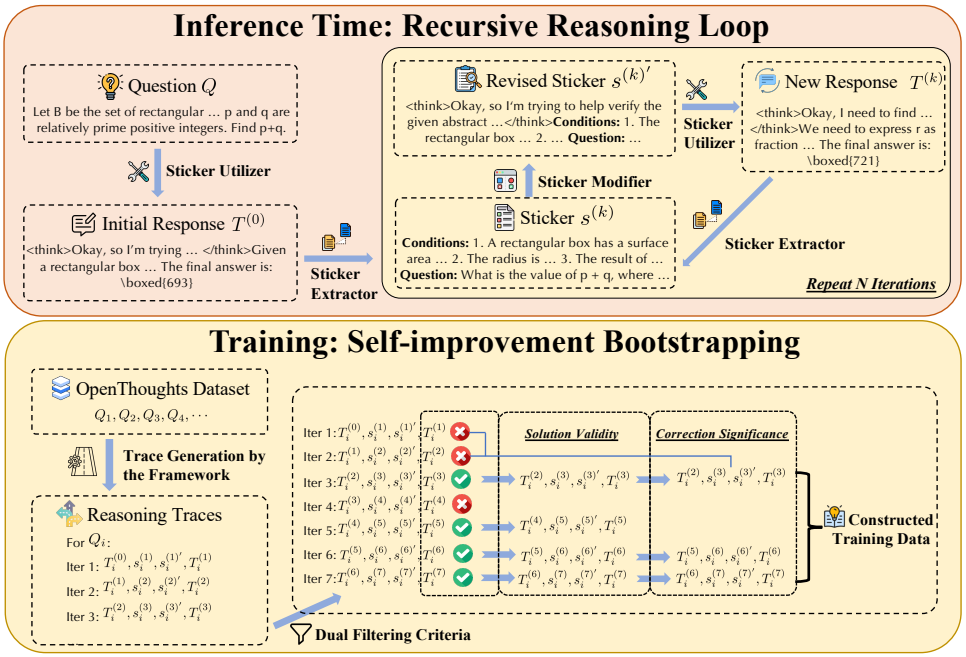

Figure 1: The overall framework of our proposed Sticker-TTS.

soning processes, which we term "*stickers*". Stickers encapsulate key conceptual anchors that guide future reasoning. At each round, we extract and refine a sticker from the previous long-form reasoning trace and embed it into the subsequent input. This approach encourages LRMs to explore alternative solutions while effectively leveraging past attempts. By using stickers as lightweight, expressive intermediates, our method enhances both reasoning robustness and efficiency.

In this paper, we propose **Sticker-TTS**, a collaborative framework designed for test-time scaling with multiple LRMs, enabling effective utilization of historical experience. The framework comprises three key components: a *Sticker Extractor*, which distills concise and relevant insights (*"stickers"*) from previous reasoning traces; a *Sticker Modifier*, which adapts these stickers to the current context; and a *Sticker Utilizer*, which integrates them to guide the model towards more effective solution strategies. During inference, these components operate iteratively, allowing the model to synthesize prior knowledge with new reasoning paths. To enhance the utility of this collaborative process, we propose a two-stage training paradigm combining knowledge distillation and self-improvement. Initially, the extractor and modifier are trained on approximately 1K distilled examples. The full framework is then used to sample collaborative reasoning trajectories, which in turn serve as new training data to iteratively refine the modules. This cycle of generation and retraining progressively enhances the model's reasoning ability, demonstrating the promise of sticker-based collaboration for scaling test-time inference.

To validate the effectiveness of Sticker-TTS, we evaluate our method on several challenging reasoning math benchmarks, including the 2024 and 2025 AIME problem sets and OlymMATH, a recently introduced Olympiad-level math benchmark. Our framework consistently outperforms competitive baselines on both benchmarks under comparable compute bugets. For example, our method achieves a 12.42% relative improvement over self-consistency on the AIME-25 using a 7B model. On the other hand, compared to models trained with reinforcement learning, our approach performs comparably or even better across multiple benchmarks—for instance, achieving a 9.79% relative improvement over Skywork-OR1 on OlymMath. Moreover, when scaling computation through multi-round reasoning, our method demonstrates further performance gains, delivering an 18.75% relative improvement over Light-R1 on AIME-25. This demonstrates the efficiency of our approach in utilizing historical experience for better test-time scaling.

2 Method

Unlike traditional test-time scaling methods, our approach focuses on refining and utilizing histor-

ical experiences. We provide an overview of our method in Section 2.1. Furthermore, we introduce the inference and training of our approaches in Section 2.2 and Section 2.3, respectively.

2.1 Overview

Our Sticker-TTS framework comprises three interrelated models: the Sticker Extractor E, the Sticker Modifier M, and the Sticker Utilizer U. These models work collaboratively through an iterative reasoning process. Given a reasoning trace, the Sticker Extractor first extracts and summarizes key reasoning steps and global strategies into a structured ***"sticker"***. The Sticker Modifier subsequently inspects this sticker for any mistakes, applying necessary corrections. Finally, the Sticker Utilizer generates an enhanced reasoning trace by integrating the modified sticker with the original question and the previous trace. We show the overall procedure in Figure 1.

Algorithm 1: Sticker-TTS Framework

Input : Question Q, Sticker Extractor E, Sticker Modifier M, Sticker Utilizer U, Max Iterations N
Output : Final Answer A_{final}

// **Initial response without sticker**
$T^{(0)}, A^{(0)} \leftarrow U(Q)$ ▷ U generate response without sticker
$TraceList \leftarrow [], AnswerList \leftarrow []$
$TraceList.append(T^{(0)})$
$AnswerList.append(A^{(0)})$
// **Recursive Reasoning Loop**
for $k \leftarrow 1$ **to** N **do**
  // **Sticker Extraction**
  $s^{(k)} \leftarrow E(T^{(k-1)}, Q)$
  // **Sticker Modification**
  $s^{(k)'} \leftarrow M(s^{(k),Q})$
  // **Trace Generation**
  $T^{(k)}, A^{(k)} \leftarrow U(s^{(k)'}, Q, A^{(k-1)})$
  $TraceList.append(T^{(k)})$
  $AnswerList.append(A^{(0)})$
end
// **Final Answer Derivation**
$A_{final} \leftarrow MajorityVote(AnswerList)$ ▷ Aggregate answers from all N traces

To obtain these components, we adopt a two-stage training strategy with distillation-guided self-improvement. At the first training stage, we initialize the framework through knowledge distillation from powerful teacher models. Specifically, we construct training data in the required format by distilling the teacher's reasoning traces, then perform fine-tuning to adapt all three models (E, M, and U) to their respective functional roles. Subsequently, we implement a self-improvement training stage where the framework autonomously generates iterative reasoning traces on open-source mathematical problems. These generated experiences undergo rigorous filtering based on solution validity and reasoning refinement trajectories, forming high-quality self-distilled data. We then conduct additional fine-tuning using this curated dataset to further enhance the models' ability in sticker extraction, error correction, and iterative optimization.

2.2 Recursive Reasoning Loop

Sticker-TTS operates through an iterative mechanism that progressively enhances reasoning quality. As illustrated in Figure 1, each iteration k (starting from $k = 1$) builds upon the previous reasoning trace $T^{(k-1)}$ and corresponding answer $A^{(k-1)}$. Notably, $T^{(0)}$ denotes the initial response generated by the Sticker Utilizer U without prior sticker integration, and $A^{(0)}$ indicates the answer extracted from $T^{(0)}$. Subsequently, our approach sequentially invokes three phases, *i.e.,* sticker extraction, sticker modification, and trace generation, within each iteration, and terminates the generation process utill meeting the stopping criterion. Below, we formalize the overall recursive process and provide the complete algorithmic flow in Algorithm 1.

Prompt for Sticker Extraction

Given the solution provided below, Generate an abstract of the key conditions that help solve the problem. The abstract should include both the key conditions and the question.
Abstract Format:
Conditions:
1. [Condition 1]
... (add more conditions as needed)
Question:
[Clearly state what is being asked.]
Requirements:
[Specify requirements that the model must meet.]
Solution to question:
[Solution]
Please provide your output strictly following ...

Sticker Extraction. The Sticker Extractor E is designed to effectively capture the primary strategy and reasoning steps while identifying weaknesses in an existing reasoning trace. It takes a reasoning trace $T^{(k-1)}$ and the corresponding question Q as input. Based on this historical trace, E extracts a structured sticker $s^{(k)}$. This sticker acts as a diagnostic summary that captures the strategic essence while pinpointing the most critical limita-

tions within the current reasoning trace. We show the utilized prompt in the following table.

Prompt for Sticker Modifier

Given a question and the abstract generated from the solution, carefully check and verify whether the extracted key conditions contain any errors in reasoning or incorrect conditions.
Step 1: Verify and refine the Conditions section.
- Conditions can come from the reasoning process.
... (Some other requirements are ommited) ...
Step 2: Verify the **Question section.**
- Ensure the question summary is concise ...
- If incorrect, provide a refined version.
Step 3: Generate the output.
- you should output your refined abstract in the following format:
Conditions:
1. [Corrected Condition 1]
... (more conditions if necessary)
Question:
[Refined question summary]
Please provide your output strictly following the step 3 without other unnecessary words.

Sticker Modification. The Sticker Modifier M examines the sticker $s^{(k)}$ to refine potential errors. According to the reasoning steps and limitations summarized in the sticker, M performs fine-grained error analysis, including computational mistakes and methodological flaws. This process generates a revised sticker $s^{(k)'}$ that incorporates corrective feedback, ensuring subsequent reasoning steps address previously identified weaknesses. We show the utilized prompt in the following table.

Prompt for Sticker Utilization

Given a question:
[Question]
Given a sticker that may be correct or incorrect:
[Sticker]
The previous answer that may be correct or incorrect:
[Answer]
Please reason step by step and put final answer in the boxed.

Sticker Utilization. The Sticker Utilizer U generate a new reasoning path $T^{(k)}$ by integrating $s^{(k)'}$ with the original question Q and the previous answer $A^{(k-1)}$. The new generated $T^{(k)}$ and $A^{(k)}$ subsequently serve as the input for the next iteration, enabling progressive refinement. We show the utilized prompt in the following table.

Stopping Criterion. The iterative loop terminates after N iterations, yielding N progressively refined reasoning traces $\{T^{(1)}, ..., T^{(k)}\}$ and corresponding answers $\{A^{(1)}, ..., A^{(k)}\}$. To derive the final answer, we aggregate all N answers through the majority vote approach.

2.3 Self-improvement Progressive Training

Although the design of our framework is clear, developing the framework's components from scratch poses significant challenges, primarily due to the need for a nuanced understanding of complex reasoning patterns. To tackle this issue, we propose a two-stage progressive training strategy. First, we utilize knowledge distillation to align the model with the target inference patterns (*i.e.,* extracting stickers, modifying stickers, and utlizing stickers). Following this, we enhance the model's performance through self-improvement bootstrapping. This approach not only streamlines the training process but also ensures a more robust understanding of the reasoning required for effective performance.

Initialization via Knowledge Distillation. The first stage establishes capacity adaptation through knowledge distillation from powerful teacher models. We construct task-aligned training data using mathematical problems marked as solvable in the OpenThoughts dataset (Team, 2025) and employ powerful `DeepSeek-R1` (Guo et al., 2025) to generate high-quality reasoning traces. For training Sticker Extractor E, we use `o3-mini` (OpenAI, 2025) to extract structured stickers from the long-form reasoning traces, which exhibit greater faithfulness compared to other reasoning models (Bao et al., 2024). Subsequently, to prepare training data for models Sticker Modifier M and Sticker Utilizer U, we simulate error-correction scenarios. Specifically, we start from the flawed reasoning traces and their corresponding stickers derived from the training data prepared for Sticker Extractor and leverage `DeepSeek-R1` as Sticker Modifier and Utilizer to examine stickers and generate refined reasoning paths. Finally, We only retain the generated data from the three models to form paired training data on the condition that the final reasoning trajectory is completely correct. Through fine-tuning on these distilled datasets, each component has preliminarily acquired its specialized capability in extraction, correction, and optimization.

Self-improvement Bootstrapping. To further enhance the model's capabilities, we enable the model to generate data autonomously and employ rigorous curation of the self-distilled training data. Leveraging the initialized framework, we iteratively generate reasoning traces on OpenThoughts

while enforcing dual filtering criteria to ensure the quality of the training data. The first criterion is *solution validity*. We preserve trajectories where the final optimized answer is correct while maintaining a 1:2 ratio between "error-to-correct" transitions (where the initial reasoning path contains errors but the final optimized answer is corrected) and "correct-to-correct" transitions (where the training path is already valid while undergoing further refinement). This ratio aligns with the statistical distribution of naturally generated reasoning paths, where correct initial attempts occur more frequently. The second criterion is *correction significance*. For selected cases where iterative refinement succeeds after previous reasoning fails, we limit the preceding two iterations to yield incorrect answers. This ensures the difficulty of the retained cases, which involve non-trivial corrections requiring sustained reasoning effort. Subsequent fine-tuning on this curated dataset enables synergistic enhancement of the framework: Sticker Extractor E improves its capacity to identify critical reasoning patterns from iterative histories, Sticker Modifier M develops robust error diagnosis through exposure to multi-failure recovery scenarios, and Sticker Utilizer U strengthens its reasoning path generation capability by integrating optimized reasoning strategies. To prevent overfitting, we limit each mathematical problem to provide at most one qualified training instance for each framework component during their respective training phases.

3 Experiments

3.1 Experimental Setup

Dataset and Benchmarks. We evaluate our method on three mathematical reasoning benchmarks: AIME 2024 (MAA, 2024), AIME 2025 (MAA, 2025), and OlymMATH-EN-EASY (Sun et al., 2025). AIME offers 30 challenging mathematical problems per year targeting academically advanced high school students. OlymMATH-EN-EASY comprises 100 Olympiad-level problems, designed to rigorously evaluate complex reasoning capabilities with verifiable numerical solutions. For model training, we use the math subset of OpenThoughts (Team, 2025), an open synthetic reasoning dataset containing $114k$ high-quality examples.

Evaluation Metrics. We employ two primary metrics: Pass@1 and Cons@N. For baseline models, Pass@1 is estimated by generating 64 responses per query using nucleus sampling with a top-p value of 0.95 and a temperature of 0.6. In our method, Pass@1 is computed directly using the answer from the final iteration. Cons@N evaluates the majority vote agreement, where baseline implementations generate N independent samples, while our method naturally accumulates N responses through iterations and performs voting across these evolution trajectories. To ensure fair comparison, we configure generation parameters consistently across models. For the `DeepSeek-R1-Distill-Qwen`[1] series, the maximum generation length is set to $32,000$ tokens. For the `Qwen2.5` series (Yang et al., 2024), the maximum generation length is configured to $5,000$ tokens.

Baselines. To ensure comprehensive evaluation, we consider LLMs trained via three approaches as baselines, including *distillation*, *multi-staged post-training featuring RL*, and *test-time scaling framework*. For the distillation approach, we adopt the `DeepSeek-R1-Distill` series as evaluation baselines. For the multi-staged post-training method with RL, we employ the `Light-R1` series (Liang Wen, 2025), `Skywork-OR1` series (He et al., 2025), and `AM-Thinking-v1` (Ji et al., 2025) as baseline LLMs. For the test-time scaling framework approach, we utilize `LeaP-T-7B` (Luo et al., 2025) and Think-Twice (Tian et al., 2025) as baselines.

Implementation Details. For data preparation, we employ `DeepSeek-R1-Distill-Qwen-7B` to sample 10 reasoning trajectories per mathematical problem in the OpenThoughts dataset. The correctness rates of these trajectories are used to estimate problem difficulty levels. During the knowledge distillation stage, we select problems with difficulty scores between 0.2 and 0.5. Responses from `DeepSeek-R1` and `o3-mini` are obtained via API calls, with sampling parameters the same as the evaluation setup. For the self-improvement bootstrapping stage, we curate more challenging data with difficulty scores ranging from 0 to 0.4. The Sticker Extractor is trained using the `Qwen2.5` series models, while both the Sticker Modifier and Sticker Optimizer utilize the `DeepSeek-R1-Distill-Qwen` series. Experiments

[1] https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-32B

Method	AIME 2024			AIME 2025			OlymMATH-EN-EASY		
	Pass@1	Cons@20	Cons@64	Pass@1	Cons@20	Cons@64	Pass@1	Cons@20	Cons@64
7B Models									
DeepSeek-R1-Distill	55.52	73.33	76.67	38.54	53.33	56.67	41.88	67.00	71.00
Light-R1	57.55	76.67	80.00	42.86	53.33	60.00	46.48	65.00	74.00
Skywork-OR1	<u>66.30</u>	76.67	83.33	**52.50**	**63.33**	63.33	57.38	<u>79.00</u>	78.00
Think-Twice	56.67	73.33	76.67	43.33	56.67	56.67	53.00	55.00	58.00
LeaP-T	64.38	<u>80.00</u>	80.00	41.25	56.67	60.00	35.95	62.00	68.00
Ours (Stage 1, N=10)	60.00	<u>80.00</u>	/	40.00	<u>60.00</u>	/	<u>61.00</u>	76.00	/
Ours (Stage 2, N=10)	**66.67**	**83.33**	/	<u>43.33</u>	**63.33**	/	**63.00**	**80.00**	/
32B Models									
DeepSeek-R1-Distill	72.60	83.33	86.67	54.37	70.00	73.33	65.34	86.00	87.00
Light-R1	76.77	86.67	86.67	64.79	73.33	76.67	75.53	89.00	92.00
Skywork-OR1	<u>80.83</u>	86.67	86.67	72.08	<u>80.00</u>	80.00	<u>85.77</u>	<u>93.00</u>	96.00
AM-Thinking-v1	**81.15**	<u>90.00</u>	90.00	**76.25**	**83.33**	83.33	**86.25**	**95.00**	96.00
Ours (Stage 1, N=10)	70.00	<u>90.00</u>	/	70.00	<u>80.00</u>	/	79.00	88.00	/
Ours (Stage 2, N=10)	76.67	**93.33**	/	<u>73.33</u>	<u>80.00</u>	/	78.00	90.00	/

Table 1: Evaluation results on three mathematical reasoning benchmarks. Note that while our method reports answers via Cons@N, its associated reasoning cost is comparable to Cons@2N. To ensure fair comparison, performance comparisons are conducted with aligned reasoning consumption. We additionally provide baseline reference performance at larger N values for context. The best and second-best results are highlighted in **bold** and <u>underlined</u>, respectively.

are conducted across two model scales: 7B and 32B. As for the SFT configuration, the maximum context length is $20,000$ tokens. The Sticker Extractor is trained with a batch size of 96 and a learning rate of 1×10^{-5}. The Sticker Modifier and Sticker Utilizer are trained with a batch size of 128 and a learning rate of 2×10^{-5}. The detailed information of SFT configuration is in Appendix A.

3.2 Main Results

Table 1 presents the performance of our method and other baselines on three representative mathematical reasoning datasets. We can make the following observations:

• *Superior Performance.* Our proposed method demonstrates better or comparable performance compared to other baselines. After the first training stage of distillation, our method already surpasses models trained through distillation and test-time scaling framework. Following the second training stage, our method outperforms most baselines, and even exceeds some models developed via multi-staged post-training featuring RL, such as `Light-R1`. Its performance is comparable to the current state-of-the-art open-source reasoning model `AM-Thinking-v1` with the metric of Cons@20. This two-stage progression indicates that the initial knowledge distillation successfully adapts the framework's components to their functional roles, while the subsequent self-improvement bootstrapping enables synergistic capability enhancement of the framework. The sustained performance gains confirm our framework's powerful capacity for reasoning path optimization and generation.

• *Scalability Across Model Sizes.* Our method demonstrates effectiveness across different model scales, achieving considerable improvements with both 7B and 32B parameter variants. This scalability demonstrates that our framework adapts well to varying model capability levels. Our framework enables effective division of labor regardless of base model size, with each component specializing in its respective task while maintaining coherent collaboration.

• *Enhanced Reasoning Efficiency*. Our method achieves substantial performance with favorable reasoning cost. After the two-stage training, our method attains superior results with only N=10 iterations, outperforming the Cons@64 performance of most baselines. Additionally, we vary the iteration number N and examine the Cons@N performance of our method on the OlympMATH-EN-EASY benchmark. As illustrated in Figure 2, our model achieves effective test-time scaling as N increases. Notably, since the Sticker Extractor operates as a model featuring short CoT, demonstrating

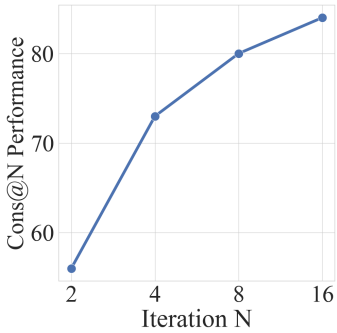

Figure 2: Cons@N performance on OlymMATH-EN-EASY across varying iteration counts N of 7B model.

Method	AIME 2024 (Cons@20)	AIME 2025 (Cons@20)
7B Models		
Ours (Stage 2, N=10)	83.33	63.33
Early Exit	70.00	56.67
Parallel Sampling (P=2, Q=5)	80.00	60.00
Parallel Sampling (P=5, Q=2)	80.00	56.67
32B Models		
Ours (Stage 2, N=10)	93.33	80.00
Early Exit	86.67	76.67
Parallel Sampling (P=2, Q=5)	93.33	73.33
Parallel Sampling (P=5, Q=2)	86.67	76.67

Table 2: Performance comparison under different iteration strategies.

substantially lower reasoning cost compared to the long-CoT Sticker Modifier and Optimizer, the total reasoning cost at N=10 remains comparable to that of Cons@20 setups for long-CoT models. This efficiency stems from our framework's enhanced capacity to perceive and learn from historical experiences. Each iteration effectively distills insights from prior attempts instead of conducting history-unaware parallel sampling.

3.3 Further Analysis

Reasoning Depth. Since our method continually refines its outputs by leveraging the history of prior responses, we can vary the number of iterations N to control the reasoning depth. We examine two strategies: *early exit* and *parallel sampling*. For early exit, an additional stopping criterion is introduced where the iteration terminates if the current response's answer matches that of the previous iteration. For parallel sampling, we partition the sampling process into P parallel chains, each executing Q iterations per query, ensuring $PQ = N$. The results of these experiments are presented in Table 2. Overall, we can have two major observations. Firstly, increasing test time enables our method to better learn from experience. While the early exit strategy reduces the average number of iterations, it appears detrimental to the refinement of stickers through deeper iterations, thereby limiting the depth of perception and learning from historical responses. Secondly, with the same reasoning costs, deeper iterations yield consistent performance gains over other methods, indicating that our method effectively leverages historical responses for sustained optimization. This suggests that the interplay among the three Sticker components progressively strengthens the consensus and accuracy of the reasoning outcome.

Method	AIME 2024 (Cons@20)	AIME 2025 (Cons@20)
7B Models		
Ours (Stage 2, N=10)	83.33	63.33
Extractor Ablation	73.33	53.33
Modifier Ablation	70.00	53.33
Full Ablation	70.00	50.00

Table 3: Ablation study in Sticker-TTS.

Method	AIME 2024		AIME 2025	
	Pass@1	Cons@20	Pass@1	Cons@20
32B Models				
DeepSeek-R1-Distill	72.60	83.33	54.37	70.00
Light-R1	76.77	86.67	64.79	73.33
Sticker Utilizer	75.68	86.67	58.54	73.33

Table 4: Evaluation results of the 32B Sticker Utilizer.

Ablation Study. To assess the effectiveness of components in our framework, we conduct ablation experiments focusing on the Sticker Extractor and Sticker Modifier. Three configurations are tested: (1) *Extractor Ablation*: Directly feeding raw reasoning traces to the Sticker Modifier without sticker extraction; (2) *Modifier Ablation*: Using unmodified stickers from the Extractor to generate new traces; (3) *Full Ablation*: Generating new traces directly from the original reasoning path without sticker involvement. As shown in Table 3, performance declines under individual component ablation, while full ablation causes the most significant degradation. This demonstrates that both components serve critical roles: the Sticker Extractor's strategy abstraction prevents the Sticker Modifier from being overwhelmed by details in reasoning traces, while the Sticker Modifier's error correction ensures sticker quality for subsequent optimization. The compounded performance loss under full abla-

tion suggests that intermediate sticker representations are likely essential for navigating the internal complexity of reasoning traces. Without structured stickers, the framework struggles to maintain strategic focus during iterative refinement, potentially propagating errors or becoming trapped in flawed reasoning patterns.

Sticker Utilizer Analysis. We conduct standalone evaluations of the 32B Sticker Utilizer after the two-stage training, without the collaboration of the other two models. As shown in Table 4, the Sticker Utilizer achieves superior performance compared to `DeepSeek-R1-Distill-32B` while matching `Light-R1` in Cons@20 metrics. This demonstrates that training models to optimize reasoning paths enhances intrinsic reasoning capabilities. Notably, while the Sticker Utilizer's Pass@1 score is lower than `Light-R1`, likely due to differences in training objectives, its Cons@20 equivalence shows that the majority vote strategy effectively overcomes the instability of single run by aggregating diverse valid trajectories. This suggests that the Sticker Utilizer possesses strong reasoning potential, and its generation stability could be enhanced with further calibration.

4 Related Work

Test-Time Scaling Techniques. Recent advances have proposed a range of decoding strategies to enhance reasoning accuracy during inference. A prominent line of work involves performing multiple sampling passes and selecting the final answer via majority voting, as exemplified by the self-consistency method (Wang et al., 2022). Building on this, confidence-weighted self-consistency (Taubenfeld et al., 2025) reduces the number of required samples by incorporating answer uncertainty. Beyond independent sampling, recent approaches leverage multiple rounds of generation informed by previous attempts, such as feeding the full prior answer back into the model (Tian et al., 2025) or adopting parallel thinking mechanisms (Luo et al., 2025). However, these long-form reasoning processes impose a significant burden on the model's long-context capabilities (Li et al., 2023), while overly brief answers limit the potential to leverage historical attempts effectively. Moreover, existing methods primarily focus on prompt design and offer limited support for iterative improvement through training. In contrast, our proposed framework introduces *stickers*, which are succinct, distilled cues extracted from extended reasoning traces, to guide the utilization of historical solutions. Furthermore, complemented by a two-stage training strategy that combines imitation learning and self-improvement, our framework enables continual enhancement of test-time reasoning performance.

Reinforcement Learning for Reasoning. With the help of RL, LRMs have achieved significant progress. Especially, OpenAI's o1 series [2], DeepSeek-R1 (Guo et al., 2025), and Kimi K1.5 (Team et al., 2025) have achieved surprising math and code performance by training with outcome-based reward on large scale. Complementary to this, methods like VC-PPO (Yuan et al., 2025), and Light-R1 (Wen et al., 2025) investigate alternative reward formulations, curriculum learning, and multi-stage training to enhance reasoning capabilities. The proliferation of open-source frameworks—including SimpleRL (Zeng et al., 2025) and STILL series work (Chen et al., 2025)-has played a vital role in replicating and scaling RL pipelines, promoting reproducibility and accelerating broader adoption. These advances collectively provide a robust foundation for efficient and reliable RL training in large models. Our approach is decoupled from the underlying model, making it pluggable with the aforementioned models to enhance their test-time scalability and performance. Additionally, our training strategy can be applied to further improve the overall performance.

5 Conclusion

In this paper, we explore how to enhance the test-time scaling performance of LRMs. We propose a novel sticker-based test-time scaling framework which consists of three modules: a *Sticker Extractor* to distill concise and relevant insights ("stickers") from previous reasoning traces; a *Sticker Modifier* to adapt these stickers to the current context; and a *Sticker Utilizer* to integrate them to guide the model towards more effective solution strategies. During inference, these components operate iteratively, allowing the model to synthesize prior knowledge with new reasoning paths. Extensive experiments validate its effectiveness, demonstrating its superiority over strong baselines.

[2] https://openai.com/o1/

Limitations

In this paper, we present a sticker-based test-time scaling framework to enhance reasoning capacities of LRMs during inference. Beyond DeepSeek-R1-Distill model, we believe our framework can be employed in broader LRMs, which have not been explored owing to the computational costs. Additionally, our method mainly focus on utilizing supervised fine-tuning (*i.e.,* RFT) to train each module in the framework. However, in the future, we can further employ RL to train the whole framework, which is an multi-agent system in essence. Limited by the computational costs, we conduct experiments on models up to 32B in size, and future work may explore validating our proposed method on even larger models.

References

Forrest Bao, Miaoran Li, Rogger Luo, and Ofer Mendelevitch. 2024. HHEM-2.1-Open.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024. Alphamath almost zero: process supervision without process. *CoRR*, abs/2405.03553.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. 2016. Training deep nets with sublinear memory cost. *CoRR*, abs/1604.06174.

Zhipeng Chen, Yingqian Min, Beichen Zhang, Jie Chen, Jinhao Jiang, Daixuan Cheng, Wayne Xin Zhao, Zheng Liu, Xu Miao, Yang Lu, et al. 2025. An empirical study on eliciting and improving r1-like reasoning models. *arXiv preprint arXiv:2503.04548*.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

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

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, Siyuan Li, Liang Zeng, Tianwen Wei, Cheng Cheng, Bo An, Yang Liu, and Yahui Zhou. 2025. Skywork open reasoner series. Notion Blog.

Yunjie Ji, Xiaoyu Tian, Sitong Zhao, Haotian Wang, Shuaiting Chen, Yiping Peng, Han Zhao, and Xiangang Li. 2025. Am-thinking-v1: Advancing the frontier of reasoning at 32b scale. *arXiv preprint arXiv:2505.08311*.

Jinhao Jiang, Zhipeng Chen, Yingqian Min, Jie Chen, Xiaoxue Cheng, Jiapeng Wang, Yiru Tang, Haoxiang Sun, Jia Deng, Wayne Xin Zhao, Zheng Liu, Dong Yan, Jian Xie, Zhongyuan Wang, and Ji-Rong Wen. 2024. Enhancing llm reasoning with reward-guided tree search. *Preprint*, arXiv:2411.11694.

Jikun Kang, Xin Zhe Li, Xi Chen, Amirreza Kazemi, and Boxing Chen. 2024. Mindstar: Enhancing math reasoning in pre-trained llms at inference time. *CoRR*, abs/2405.16265.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2023. Loogle: Can long-context language models understand long contexts? *arXiv preprint arXiv:2311.04939*.

Fenrui Xiao Xin He Qi An Zhenyu Duan Yimin Du Junchen Liu Lifu Tang Xiaowei Lv Haosheng Zou Yongchao Deng Shousheng Jia Xiangzheng Zhang Liang Wen, Yunke Cai. 2025. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Tongxu Luo, Wenyu Du, Jiaxi Bi, Stephen Chung, Zhengyang Tang, Hao Yang, Min Zhang, and Benyou Wang. 2025. Learning from peers in reasoning models. *arXiv preprint arXiv:2505.07787*.

MAA. 2024. American Invitational Mathematics Examination - AIME 2024.

MAA. 2025. American Invitational Mathematics Examination - AIME 2025.

OpenAI. 2024a. Learning to reason with llms. Accessed: 2025-05-03.

OpenAI. 2024b. Openai o1 system card. Accessed: 2025-05-03.

OpenAI. 2025. Openai o3 mini. Accessed: 2025-05-03.

Haoxiang Sun, Yingqian Min, Zhipeng Chen, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, Lei Fang, and Ji-Rong Wen. 2025. Challenging the boundaries of reasoning: An olympiad-level math benchmark for large language models. *CoRR*, abs/2503.21380.

Amir Taubenfeld, Tom Sheffer, Eran Ofek, Amir Feder, Ariel Goldstein, Zorik Gekhman, and Gal Yona. 2025. Confidence improves self-consistency in llms. *arXiv preprint arXiv:2502.06233*.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. 2025. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*.

Open Thoughts Team. 2025. Open Thoughts.

Xiaoyu Tian, Sitong Zhao, Haotian Wang, Shuaiting Chen, Yunjie Ji, Yiping Peng, Han Zhao, and Xiangang Li. 2025. Think twice: Enhancing llm reasoning by scaling multi-round test-time thinking. *arXiv preprint arXiv:2503.19855*.

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*.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, et al. 2025. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond. *arXiv preprint arXiv:2503.10460*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

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

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

Yufeng Yuan, Yu Yue, Ruofei Zhu, Tiantian Fan, and Lin Yan. 2025. What's behind ppo's collapse in long-cot? value optimization holds the secret. *arXiv preprint arXiv:2503.01491*.

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*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *CoRR*, abs/2303.18223.

A SFT Configuration

We utilize the huggingface Transformers (Wolf et al., 2019) to implement our experiments, using Flash Attention (Dao et al., 2022) and DeepSpeed ZeRO Stage 3 to optimize the training efficiency. We employ AdamW optimizer (Loshchilov and Hutter, 2019) with $\beta_1 = 0.9$ and $\beta_2 = 0.95$, and use the cosine learning rate scheduler. We use BFloat16 mixed precision, with a warmup ratio of 0.1 and a weight decay of 0.1 to ensure training stability. To enhance computational efficiency, we apply gradient checkpointing strategy (Chen et al., 2016).