CONTINUAL TEST-TIME LEARNING WITH EVOLUTION STRATEGIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models increasingly operate in open-ended environments, where they are expected to continually adapt to evolving tasks and distributions, often without access to external supervision. However, existing continual learning approaches often rely on external supervision, such as labeled data, demonstrations, or reward signals, which can be costly and unavailable in practice. In contrast, repeated test-time adaptation using self-generated supervision can lead to policy collapse, degraded reasoning, and catastrophic forgetting. In this paper, we propose **Evolution Strategy Reinforcement Learning (ESRL)**, a continual test-time learning framework designed to mitigate catastrophic forgetting during label-free continual adaptation. Specifically, ESRL constructs a population of locally perturbed policies around the current model to explore multiple neighboring parameter regions throughout adaptation, and then samples trajectories from those regions. Unlike trajectory-space exploration from a single evolving policy, this parameter-space population maintains diverse sources of self-generated supervision even when the central policy begins to drift. The resulting trajectories are aggregated into label-free group-relative learning signals to update the central student. Through extensive experiments on single-task test-time adaptation and multi-task continual learning across mathematical reasoning, general knowledge, and medical domains, ESRL achieves relative gains of up to 31.7% over the base model and 16.6% over the second-best method. Notably, ESRL substantially reduces catastrophic forgetting and maintains the strongest final performance at the end of continual learning.

1 INTRODUCTION

Large Language models (LLMs) are deployed as autonomous agents that increasingly interact with users, tools, and environments over extended periods of time [3, 33]. Unlike conventional offline settings, these open-ended environments continuously introduce new tasks, evolving objectives, and previously unseen problems. To remain effective after deployment, language agents must continually improve their reasoning abilities from ongoing experience rather than relying solely on static pre-training datasets [22, 4, 34, 13].

Continual learning provides a natural way to support such adaptation by enabling models to acquire new capabilities while retaining previously learned ones [27, 32]. However, existing approaches primarily depend on external supervision, such as labeled data [27], demonstrations [22], or reward signals [27]. As language agents are deployed at scale, obtaining such supervision for every newly encountered task becomes increasingly impractical [4]. This naturally raises a fundamental question: Can a language model continually improve its reasoning at test time, without external supervision? Test-time learning provides a promising step toward this goal by allowing models to adapt using only unlabeled data encountered after deployment. Recent work has demonstrated that self-generated supervision can drive reasoning capabilities without requiring external labels [36, 35, 31, 10]. However, self-improvement introduces a new challenge. When a model repeatedly learns from its own outputs, future updates become increasingly dependent on previous generations. Over time, the resulting self-reinforcing feedback loop can amplify erroneous pseudo-labels, reduce reasoning diversity [10], and progressively collapse exploration toward a narrow set of reasoning behaviours [35]. As adaptation continues, previously acquired capabilities may be overwritten, leading to severe catastrophic forgetting [36].

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

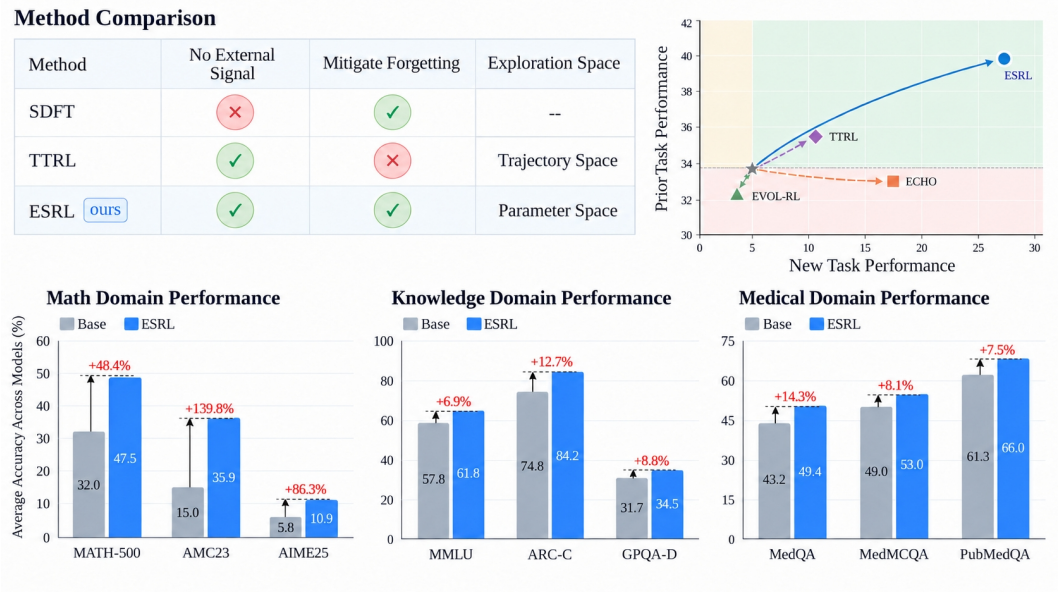

Figure 1: ESRL comparisons, continual learning performance, and overall performance summary. Top left: Comparison with existing methods for continual test-time learning across three key dimensions. Top right: Continual learning performance after training on MATH-500. Bottom: ESRL consistently improves the base models across domain-specific benchmarks and in-domain transfer tasks. All results are reported as accuracy (%).

Consequently, the central challenge is no longer simply whether a model can learn without supervision, but *whether it can continually improve its reasoning while preserving the diversity necessary for sustained exploration and retaining previously acquired capabilities*.

To address this challenge, we propose Evolution Strategy Reinforcement Learning (ESRL), a continual test-time learning framework that performs policy-level exploration without external supervision. Instead of relying on a single evolving policy to generate self-supervision, ESRL constructs a population of locally perturbed policies around a shared anchor policy. Each perturbed policy independently generates a solution to the same prompt, enabling exploration across multiple nearby regions of parameter space and producing diverse reasoning trajectories. To maintain both stability and continual adaptation, ESRL keeps the anchor fixed within each block and periodically refreshes it with the current student at block boundaries, enabling exploration to remain centered around an evolving yet stable reference. By maintaining policy-level diversity throughout adaptation and transferring robust consensus signals to a continuously updated student model, ESRL enables effective continual test-time learning while mitigating catastrophic forgetting.

Extensive experiments across four backbone models and nine benchmarks show that ESRL consistently improves test-time adaptation across mathematical reasoning, general knowledge, and medical reasoning tasks. ESRL improves average accuracy by up to 31.7% over the base model and 16.6% over the second-best method. In continual learning, ESRL better retains previously acquired capabilities and achieves the best final performance across all three domains, outperforming the second-best method by up to 90% at the end of training.

To summarize, our contributions are outlined below:

- We propose ESRL, a continual test-time learning framework that performs policy-level exploration without external supervision. ESRL constructs a population of locally perturbed policies and learns from their collective self-generated feedback.
- ESRL introduces three key designs for stable and effective adaptation: *parameter-space exploration* to broaden the reachable reasoning space, *a group-relative learning objective* to aggregate signals across the policy population, and *periodic anchor block updates* to consolidate newly acquired capabilities while preserving a stable reference policy.
- Extensive experiments show that ESRL achieves sustained reasoning gains under label-free adaptation and consistently maintains stronger performance throughout long-horizon continual learning.

2 Related Work

2.1 Test Time Learning

Test-time learning adapts a model directly using unlabeled data encountered after deployment. Recent work has extended this idea to large language models through self-generated supervision. TTRL [36] uses majority voting over sampled responses to construct label-free rewards for test-time reinforcement learning. Subsequent work has mainly focused on improving the reliability and diversity of these self-generated rollouts. ETTRL [10] introduces entropy-guided tree rollouts to improve exploration and uses advantage reshaping to reduce early overconfidence. EVOL-RL [35]introduces novelty-aware rewards to counter the loss of reasoning diversity caused by majority-only optimization. ECHO [31] further combines entropy and confidence during rollout generation and optimization to reduce rollout collapse and noisy pseudo-label reinforcement. More recent methods further target the fragility of majority-based supervision. Hi-TTRL [28] regulates rollout consensus to avoid unreliable or vanishing learning signals, and TTPO [24] applies different objectives to trajectories that agree or disagree with the pseudo-label.

Despite these advances, existing methods still construct self-supervision primarily from trajectories generated by the current policy. Once the policy drifts toward a narrow or erroneous region, its self-generated signals may reinforce the same behaviour. Moreover, these methods mainly focus on the current test distribution and do not explicitly preserve prior capabilities under sequential adaptation. ESRL addresses both limitations through parameter-space population exploration for more diverse and stable continual test-time learning.

2.2 Continual Learning

Continual learning aims to acquire new capabilities while preserving previously learned ones [27, 32]. Existing approaches mainly mitigate catastrophic forgetting through replay, regularisation, or parameter isolation. Continual fine-tuning can cause LLMs to forget previously learned knowledge and reasoning skills [11]. Replay-based methods reduce forgetting by training again on data from earlier tasks, while O-LoRA and LLaMA-Pro limit interference by separating or expanding the trainable parameters [25, 26]. Transfer-Selective Replay [13] further selects which past examples to replay based on their compatibility with the current task. Recent studies suggest that reinforcement-based adaptation can preserve previously learned capabilities better than supervised fine-tuning during sequential post-training [8]. This advantage has been linked to the use of on-policy data and the smaller distributional shift it induces [23, 1]. Building on this idea, SDFT [22] generates on-policy training signals from expert demonstrations through self-distillation, substantially reducing forgetting across sequential tasks. However, it still assumes access to demonstrations for each new task.

Despite progress in capability retention, existing continual learning methods generally rely on stored data, demonstrations, or other external supervision when new tasks arrive. They therefore do not directly address continual adaptation from unlabeled test-time data, where the model must generate its own learning signals while avoiding catastrophic forgetting. ESRL bridges this gap by enabling label-free continual test-time adaptation while maintaining diverse policy-level supervision to preserve previously acquired capabilities.

2.3 Parameter-space Exploration

Most reinforcement learning methods for LLMs explore by sampling different trajectories from the same policy. Parameter-space exploration instead perturbs the model parameters, allowing different policies to produce different behaviours [21, 17]. Evolution Strategies (ES) extend parameter-space exploration by evaluating a population of perturbed policies and using their fitness to update the model [20]. Recent work shows that ES can scale to billion-parameter LLMs [18]. ESRL combines these two forms of exploration. It uses ES-style parameter perturbations to generate diverse policies, while their trajectories are converted into group-relative learning signals to update the student with an advantage-weighted objective.

3 Method

In this section, we introduce **Evolution Strategy Reinforcement Learning (ESRL)**, a continual test-time learning framework. At a high level, ESRL consists of two phases. In Phase 1 (§3.1), ESRL performs parameter-space population exploration and assigns rewards to the resulting roll-

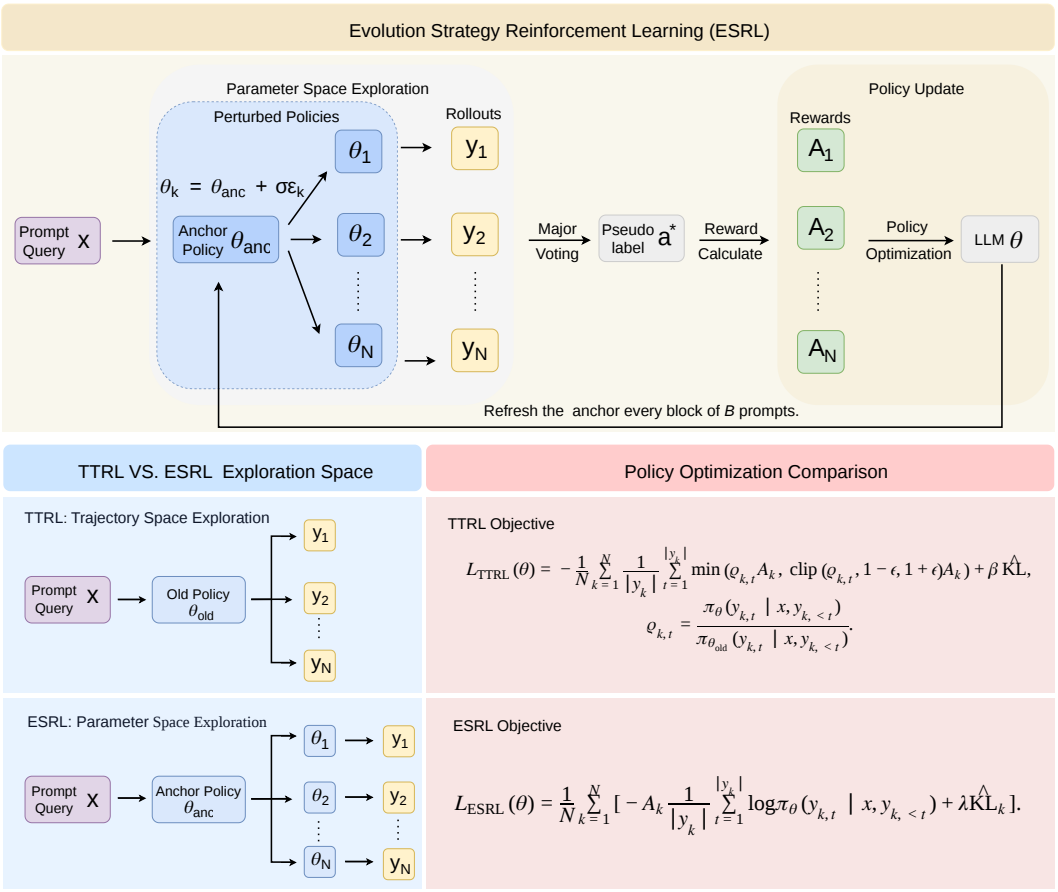

Figure 2: Overview of ESRL. ESRL first constructs a population of perturbed policies around a shared anchor and generates one rollout from each policy. Majority voting provides label-free rewards, which are converted into group-relative advantages. The student is then updated with an advantage-weighted log-likelihood objective. The anchor remains fixed within each block of B prompts and is refreshed from the latest student at the next block boundary.

outs. In Phase 2 (§3.2), ESRL converts these group-relative rewards into student updates using an advantage-weighted log-likelihood objective within each block.

3.1 Parameter-space Population Exploration

Let θ denote the trainable parameters of the student. We draw N independent perturbation directions $\varepsilon_1, \ldots, \varepsilon_N$, where $\varepsilon_k \sim \mathcal{N}(0, I)$, and form N perturbed policies:

$$\theta_k \;=\; \theta_{\text{anc}} + \sigma\varepsilon_k, \qquad k = 1, \ldots, N, \tag{1}$$

where σ is the perturbation radius. For the unlabeled prompt x, rollout k is then generated by its corresponding policy:

$$y_k \;\sim\; \pi_{\theta_k}(\cdot \mid x). \tag{2}$$

Thus, the N rollouts are generated by N distinct perturbed policies rather than repeatedly sampled from a single policy.

Label-free reward and group-relative advantage Given the rollout group $\{y_k\}_{k=1}^{N}$ for a prompt x, we extract $a_k = \text{Ans}(y_k)$ and choose the majority answer as the pseudo label:

$$a^\star \;\in\; \arg\max_a \big|\{k : a_k = a\}\big|, \qquad r_k \;=\; \mathbf{1}[a_k = a^\star]\,. \tag{3}$$

The rewards are standardized within the group to obtain the group-relative advantages:

$$A_k \;=\; \frac{r_k - \bar{r}}{\text{Std}_{j=1,\ldots,N}(r_j) + \epsilon}, \qquad \bar{r} \;=\; \frac{1}{N}\sum_{j=1}^{N} r_j, \tag{4}$$

where $\epsilon > 0$ is a small numerical constant. We retain only rollout groups that contain a clear majority without complete agreement. The majority answer must be supported by at least two rollouts, while its fraction in the group must remain below $p_{\max}$.

3.2 Group-Relative Student Update

For each retained prompt x with rollout advantage pairs $\{(y_k, A_k)\}_{k=1}^{N}$, we update the student using an advantage-weighted log-likelihood objective:

$$\mathcal{L}(\theta) \;=\; \frac{1}{N}\sum_{k=1}^{N}\left[-A_k \underbrace{\frac{1}{|y_k|}\sum_{t=1}^{|y_k|}\log \pi_\theta(y_{k,t} \mid x, y_{k,<t})}_{\text{mean per-token log-likelihood}} + \lambda \widehat{\mathrm{KL}}_k\right], \tag{5}$$

where λ controls regularisation toward the frozen initial policy π_{θ_0}. Positive advantages increase the likelihood of consensus-supported trajectories, while negative advantages suppress inconsistent ones. We use the non-negative k_3 estimator for the KL term:

$$\begin{aligned} \widehat{\mathrm{KL}}_k &= \frac{1}{|y_k|}\sum_{t=1}^{|y_k|}\left(e^{\delta_{k,t}} - \delta_{k,t} - 1\right), \\ \delta_{k,t} &= \log \pi_{\theta_0}(y_{k,t} \mid x, y_{k,<t}) - \log \pi_\theta(y_{k,t} \mid x, y_{k,<t}). \end{aligned} \tag{6}$$

Block-wise anchoring The rollout population should remain centred on a stable policy while still tracking the evolving student. Re-anchoring after every prompt would make batched population generation inefficient, whereas keeping the same anchor throughout training would cause the population to explore around an outdated policy. We therefore process prompts in blocks of B examples. At the beginning of each block, we take a snapshot of the current student and use it as the anchor to generate N perturbed models. The anchor remains fixed within the block and is refreshed to the latest student at the next block boundary.

Memory-efficient perturbations. Within each block, the same N perturbation directions are shared across all B prompts, allowing each population member to generate its B trajectories in a single batched call. To avoid materialising N model copies, each perturbation is represented by an integer seed s_k. For population member k, we deterministically regenerate $\varepsilon(s_k)$ and temporarily set the trainable parameters to $\theta_{\text{anc}} + \sigma\varepsilon(s_k)$ during generation. The parameters are then restored to the anchor before moving to the next member. This avoids storing N perturbed model copies simultaneously. At any time, we keep only the anchor parameters and a single temporary perturbation buffer, while each population member is represented by a lightweight random seed.

4 Experiment

We evaluate our method under single-task test-time adaptation §4.2 and multi-task continual learning §4.3.

4.1 Experimental Setup

Models To evaluate the generalization of ESRL across different backbone models, we conduct experiments using models of various scales from different families. We use Qwen3-0.6B, Qwen3-4B, Qwen3-8B [29], and Llama-3.2-3B [14].

Baselines We compare against the following representative methods: TTRL [36], ECHO [31], and EVOL-RL [35]. We further include two additional baselines adapted to our setting: ES [18] and SFT [15]. We report Pass@n for all methods to ensure a consistent evaluation protocol. Additional baseline explanations and implementation details are provided in Appendix A.

Benchmarks We evaluate ESRL across three domains. For each domain, we use one benchmark as the source task for adaptation and source-benchmark evaluation, and additional benchmarks from the same capability domain to measure in-domain generalization. We further use benchmarks from other capability domains to assess out-of-domain generalization. For mathematics, we adapt the model on MATH-500 [9] and evaluate on held-out MATH-500 examples, with AMC23 [12] and AIME25 [30] used to test in-domain generalization. For general knowledge, we adapt on MMLU [5] and evaluate on held-out MMLU examples, with ARC-Challenge [2] and GPQA-Diamond [19] as in-domain generalization benchmarks. For the medical domain, we adapt on the MedQA-USMLE

Table 1: Pass@1 results for source-benchmark evaluation and in-domain generalization across different backbone models.

Model	Method	Math			General Knowledge			Medical			Avg.
		MATH-500	AMC23	AIME25	MMLU	ARC-C	GPQA-D	MedQA	MedMCQA	PubMedQA	
Qwen3-0.6B	Base model	5.0	6.0	0.0	44.0	65.5	26.3	23.5	37.0	45.5	28.1
	TTRL	10.0	13.3	3.3	44.5	70.0	25.8	32.5	33.5	52.5	31.7
	ES	6.0	6.0	0.0	47.0	68.5	27.3	28.0	37.5	45.5	29.5
	SFT	0.0	0.0	0.0	43.0	60.5	28.3	22.5	23.0	**60.0**	26.4
	ECHO	17.0	9.9	3.3	46.0	69.5	23.7	18.5	37.5	45.5	30.1
	EVOL-RL	4.0	14.9	0.0	46.5	69.5	27.3	22.5	32.0	56.0	30.3
	ESRL	**27.0**	**19.3**	**6.7**	**49.5**	**70.5**	**30.8**	**34.0**	**38.0**	57.0	**37.0**
Qwen3-4B	Base model	55.0	24.1	10.0	69.0	88.0	33.8	52.0	55.5	67.0	50.5
	TTRL	70.0	45.8	13.3	67.5	90.0	35.4	51.0	**58.0**	62.5	54.8
	ES	53.0	30.1	3.3	68.0	88.0	34.8	50.0	55.5	66.0	49.9
	SFT	8.0	10.3	0.0	60.0	84.5	26.8	50.5	**58.0**	68.0	40.7
	ECHO	71.0	47.7	13.3	70.0	89.5	34.3	51.0	56.5	62.5	55.1
	EVOL-RL	68.0	48.2	3.3	65.5	89.5	**36.4**	50.0	56.0	65.0	53.5
	ESRL	**74.0**	**51.8**	**16.7**	**71.5**	**98.0**	35.9	**58.5**	**58.0**	**69.0**	**59.3**
Qwen3-8B	Base model	65.0	27.3	13.3	**73.5**	90.5	36.9	57.5	60.0	71.5	55.1
	TTRL	72.0	55.4	**16.7**	69.5	92.0	40.9	**62.0**	61.5	71.5	60.2
	ES	61.0	20.5	3.3	**73.5**	91.0	37.4	58.5	60.0	70.5	52.9
	SFT	13.0	2.4	0.0	67.5	89.5	35.4	58.5	**62.5**	71.5	44.5
	ECHO	72.0	45.8	10.0	72.0	91.5	40.4	58.5	**62.5**	71.5	58.2
	EVOL-RL	71.0	55.4	13.3	71.0	92.5	38.4	60.5	62.0	71.5	59.5
	ESRL	**78.0**	**57.5**	**16.7**	72.5	**100.0**	**41.4**	60.5	**62.5**	**72.5**	**62.4**
Llama-3.2-3B	Base model	3.0	2.4	0.0	44.5	55.0	**29.8**	40.0	43.5	61.5	31.1
	TTRL	7.0	3.2	**3.3**	53.0	**71.0**	24.2	44.0	49.0	65.0	35.5
	ES	0.0	0.0	0.0	43.5	55.8	27.8	41.5	46.0	62.5	30.8
	SFT	0.0	0.0	0.0	20.5	27.0	25.8	12.5	19.5	53.0	17.6
	ECHO	4.0	3.6	0.0	**55.5**	69.0	28.3	**46.0**	52.0	**65.5**	36.0
	EVOL-RL	3.2	1.2	0.0	53.0	69.0	26.3	43.5	49.5	63.0	34.3
	ESRL	**11.0**	**14.8**	**3.3**	53.5	68.5	**29.8**	44.8	**53.5**	**65.5**	**38.3**

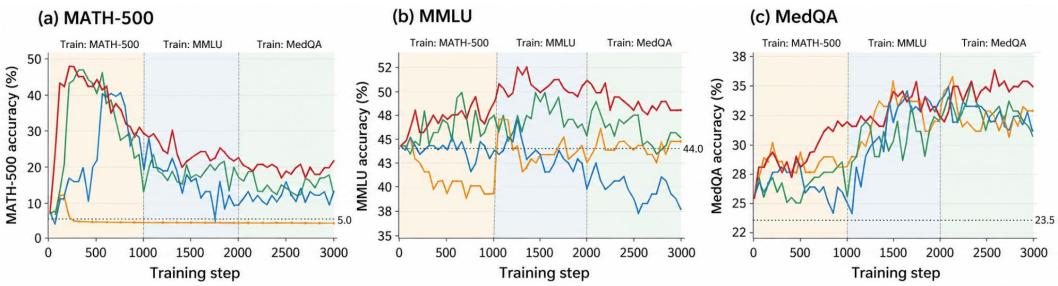

Figure 3: Sequential training of Qwen3-0.6B on MATH-500, MMLU, and MedQA. Pass@1 is reported across 3000 training steps for (a) MATH-500, (b) MMLU, and (c) MedQA.

training set [6] and evaluate on the MedQA test set, with MedMCQA [16] and PubMedQA [7] used for in-domain generalization.

4.2 Main Results

Table 1 summarizes the main results across different model backbones and domains. Across all four backbone models, **ESRL consistently outperforms the baselines in average accuracy.** The largest relative gains reach 31.65% over the base model and 16.61% over the second-best method. The gains are also consistent across training domains and model scales: ESRL improves upon or matches the base model on 35 of 36 backbone–benchmark combinations and achieves the best or tied-best performance on 29 of 36. Additionally, **the improvement extends beyond the source benchmarks.** Averaged across the four backbones, ESRL improves source-benchmark accuracy from 44.33 to 52.90 and in-domain generalization accuracy from 39.60 to 47.40, corresponding to relative gains of 19.32% and 19.71%, respectively. The gains are particularly strong in the Math domain, where average accuracy increases from 17.59 to 31.40, with ESRL achieving the best or tied-best result on all 12 Math evaluations.

4.3 Continual Test-Time Learning

In this section, we investigate a long-horizon continual learning setting to examine whether the degradation observed above accumulates over sequential adaptation. Following the protocol of Shenfeld et al. [22], a single model is trained on three tasks in sequence, while all tasks are evaluated continuously throughout training.The model is trained sequentially on MATH-500 [9], MMLU [5], and MedQA [6].

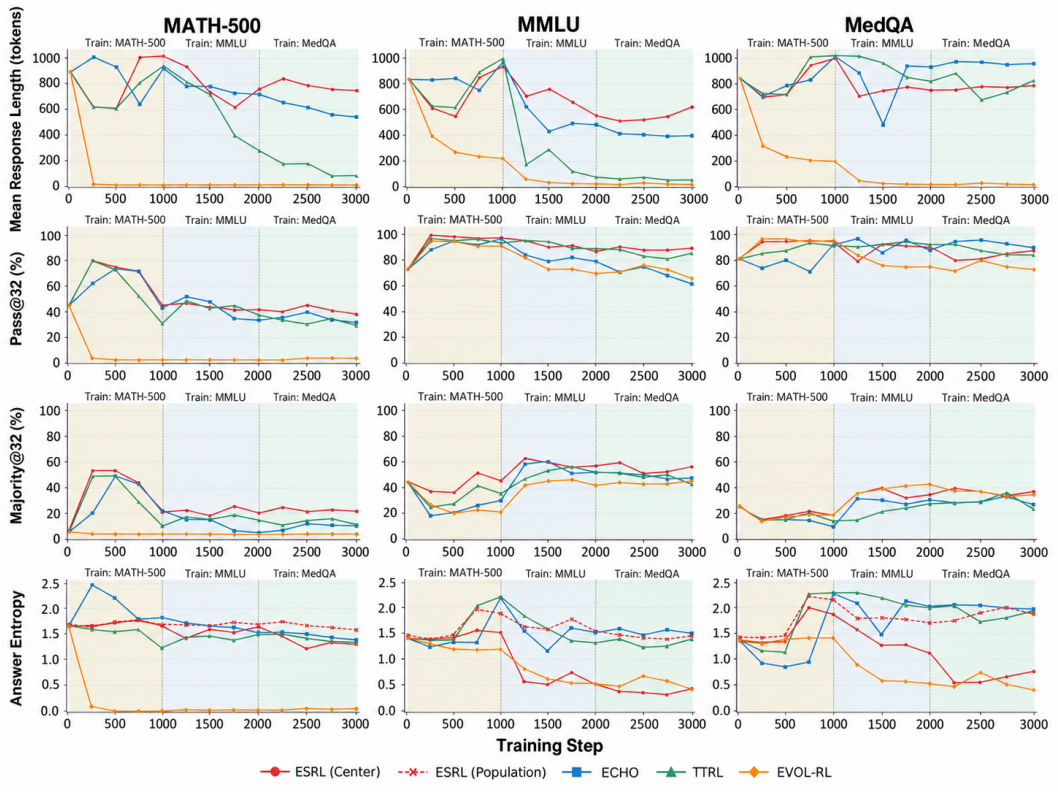

Figure 4: Training dynamics during continual learning. The first row shows the mean response length, the second row shows Pass@32, the third row shows Majority@32, where ESRL reports population-level majority voting, and the fourth row shows answer entropy. The three columns correspond to MATH-500, MMLU, and MedQA, respectively. All metrics are evaluated every 250 training steps. Detailed definitions of these metrics are provided in Appendix A.2

As shown in Figure 3, ESRL progressively accumulates new capabilities while better preserving those acquired earlier in the sequence. On MATH-500, which is learned first, ESRL shows substantially slower degradation after training shifts to MMLU and MedQA. On MMLU, ESRL improves during the second training stage and retains most of this gain after the model moves to MedQA, whereas the baselines exhibit larger degradation. Finally, on MedQA, ESRL continues to acquire the new capability during the last stage and achieves the strongest final performance. Together, these results show that ESRL better balances the acquisition of new capabilities with the retention of previously learned ones during long-horizon sequential adaptation.

5 Analysis

5.1 Why Policy-Level Exploration Works

To better understand the mechanism underlying ESRL, we investigate why policy-level exploration can better mitigate catastrophic forgetting and have better performance in continual learning tasks than trajectory-level exploration methods.

Takeaway 1: ESRL concentrates the center policy without policy collapse. The results show clear differences in how the compared methods shape their output distributions. EVOL-RL has very low answer entropy, suggesting that its output distribution becomes highly concentrated. However, its response length also drops sharply during training, together with a clear decrease in Pass@32. This pattern suggests policy collapse: the model becomes concentrated on a small set of outputs and increasingly confined to a narrow local region of the reasoning space, reducing its ability to explore alternative reasoning modes during subsequent tasks and resulting in worse continual-learning performance.

TTRL and ECHO show the opposite behavior. Their relatively high answer entropy indicates a more diffuse output distribution, while high Pass@32 suggests that they preserve broad reasoning

coverage, so correct solutions can still be found when multiple rollouts are sampled. However, the much lower Pass@1 suggests that correct solutions are often assigned relatively low probability. As a result, the model may reach the correct answer after several rollouts, while being much less likely to produce it in a single attempt. This is less useful in verifier-free settings, where generating more candidates does not by itself tell us which one is correct.

ESRL avoids both extremes. Its center policy has lower answer entropy than TTRL and ECHO, while its response length does not show the sharp contraction observed in EVOL-RL. This suggests that ESRL concentrates probability mass on a smaller set of useful outputs without collapsing the center policy, which is also consistent with its stronger Pass@1.

Takeaway 2: Population exploration preserves broad reasoning coverage. ESRL maintains broader exploration through its population of locally perturbed policies. As shown in the last row of Fig. 4, ESRL (Population) maintains substantially higher answer entropy than the center policy, indicating that the perturbed policies cover a broader set of outputs. The higher population entropy suggests that ESRL preserves a broader reasoning space during continual learning. This broader coverage makes the model less likely to become trapped in a narrow local region of the reasoning space, helping preserve alternative reasoning modes across tasks and alleviate catastrophic forgetting.

Takeaway 3: Population consensus provides a reliable learning signal. As shown in Fig. 4, ESRL Population-Maj@32 remains relatively high, particularly in the later stages of training. This shows that, despite broader exploration, the perturbed population can still form a reliable consensus around the correct answer. Since majority voting is used to construct the learning signal, this provides ESRL with effective supervision without an external verifier. This becomes especially important in later stages of continual learning, where population-level exploration keeps alternative reasoning modes reachable, while a high Maj@32 ensures that these diverse rollouts can still produce a useful dominant signal for continued optimization.

5.2 Useful Stochasticity vs. Signal Dilution

We further investigate why MATH-500 accuracy first improves rapidly and then declines during the MATH training stage of continual test-time learning. This behavior is closely related to how self-generated signals are aggregated. Since continual test-time learning has no ground-truth labels, supervision comes entirely from the model's own rollouts. Some of these signals can reveal useful reasoning modes that are not yet captured by the current policy. With a small batch, such sparse informative signals have a larger influence on each update, enabling rapid early improvement. As the batch size increases, these signals are averaged with more weak or conflicting updates, leading to more stable but weaker adaptation. However, the higher variance of small batches also makes them more sensitive to noisy pseudo-labels. As training continues and useful signals become less frequent, noise can gradually dominate the updates and cause performance to decline. As shown in Fig. 5, with a batch size of 2, MATH-500 accuracy quickly rises to around 45% before gradually decreasing to 27%. A batch size of 10 shows a similar pattern, but with a lower peak of around 20%, while a batch size of 20 produces a much smoother increase and reaches only around 10%. TTRL shows a similar batch-size-dependent pattern. Overall, batch size plays an important role in continual test-time learning. Small batches preserve sparse useful exploratory signals and enable faster adaptation, but are more sensitive to noise, while larger batches provide more stable updates at the cost of diluting informative signals.

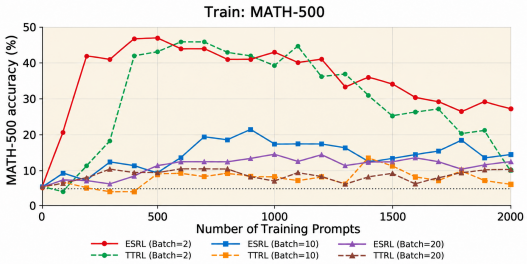

Figure 5: Comparison of Pass@1 on MATH-500 under different batch sizes.

6 Ablation Study

Block Size We further study the effect of the block size B on Qwen3-0.6B across all three domains. As shown in Table 2, performance is relatively robust to B, with the default setting $B = 10$ achieving or matching the best result on six of the nine benchmarks. Performance degrades at $B = 50$, as the anchor remains fixed while the student continues to update. This causes the population to explore around an increasingly stale policy and increases the mismatch between the generated trajectories and the current student. When $B = 2$, the anchor is refreshed after every training step. However,

this reduces batching efficiency because each perturbed policy processes only a few prompts before the population is reconstructed. Overall, a moderate block size provides a better balance between computational efficiency, exploration stability, and alignment with the evolving student.

Table 2: Pass@1 under different block sizes B using Qwen3-0.6B.

B	Math			General Knowledge			Medical			Avg.
	MATH-500	AMC23	AIME25	MMLU	ARC-C	GPQA-D	MedQA	MedMCQA	PubMedQA	
2	23.0	16.9	**6.7**	48.0	67.0	30.3	31.0	**41.0**	48.0	34.7
10	**27.0**	19.3	**6.7**	**49.5**	**70.5**	30.8	**34.0**	38.0	**57.0**	**37.0**
20	25.0	15.3	**6.7**	**49.5**	65.5	**32.3**	30.5	35.0	54.5	34.9
50	17.0	**22.9**	0.0	46.0	63.5	27.3	**34.0**	39.0	47.5	33.0

Group Size. We further study the effect of the rollout group size N on the Math domain using Qwen3-4B. We compare $N \in \{16, 32, 64\}$ while keeping all other settings unchanged. As shown in Table 3, the performance of ESRL remains relatively stable across different group sizes, indicating that the method is robust to this hyperparameter. A larger group size can provide broader exploration and a more reliable group-level learning signal. However, it can also introduce more noise into the pseudo-labeling process, so increasing N does not necessarily lead to better performance.

Table 3: Pass@1 under different rollout group sizes N using Qwen3-4B.

Group Size N	Math			Avg.
	MATH-500	AMC23	AIME25	
16	74.0	51.8	**16.7**	47.5
32	72.0	50.6	**16.7**	46.4
64	**75.0**	**53.0**	**16.7**	**48.2**

Loss Function We investigate how population-generated trajectories should be used to update the student. We keep the policy-level exploration and rollout generation of ESRL unchanged, and compare our advantage-weighted log-likelihood objective with a GRPO-style objective. Since each rollout in ESRL is generated by its own perturbed policy, the GRPO variant uses the corresponding π_{θ_k} in the importance ratio. The two variants therefore use the same rollouts and group-relative advantages, differing only in the student update: our objective directly optimizes trajectory likelihood, whereas GRPO uses a clipped importance-ratio objective. The advantage-weighted loss performs better across both in-domain and out-of-domain evaluation. Averaged over the three Math benchmarks, performance improves from 42.8 to 47.5, while the average across the six out-of-domain benchmarks increases from 48.9 to 52.1. It improves on eight of the nine benchmarks and matches GRPO on AIME25, suggesting that direct advantage-weighted likelihood optimization is better suited to the population-generated trajectories in ESRL.

Table 4: Pass@1 under different optimization objectives using Qwen3-4B.

Loss Function	Math			General Knowledge			Medical			Avg.
	MATH-500	AMC23	AIME25	MMLU	ARC-C	GPQA-D	MedQA	MedMCQA	PubMedQA	
GRPO-Style Clipped Loss	67.0	44.6	**16.7**	65.5	87.5	29.3	49.0	54.0	61.5	52.8
Advantage-Weighted Loss	**74.0**	**51.8**	**16.7**	**68.5**	**89.0**	**33.3**	**53.0**	**56.0**	**66.0**	**56.5**

7 CONCLUSION AND LIMITATION

In this paper, we introduced ESRL, a continual test-time learning framework that combines parameter-space population exploration with label-free group-relative learning. By maintaining a population of locally perturbed policies around an evolving anchor, ESRL enables broader exploration while preserving a stable center policy for continual adaptation. Experiments across diverse reasoning domains show that ESRL consistently improves test-time learning and better retains previously acquired capabilities during long-horizon continual learning. Further analysis suggests that its advantage comes from maintaining broad reasoning coverage without collapsing the center policy. Overall, our results highlight policy-level population exploration as a promising approach to continual test-time adaptation without external supervision.

Limitation. Although ESRL is evaluated across four backbone models and nine benchmarks, broader validation on more diverse and high-stakes tasks would further strengthen its empirical robustness. We leave this direction for future work.

AI USE STATEMENT

In this work, we used generative AI tools for language editing, improving clarity and presentation, and assisting with code debugging and formatting. We did not use generative AI to generate experimental results or fabricate data, and other disclosure categories are not applicable. All AI-assisted text and code were reviewed, verified, and revised by the authors. We take full responsibility for the final content of this work, including all text, claims, code, and results produced with the aid of generative AI.

ETHICS STATEMENT

This work does not involve human subjects, personally identifiable information, or sensitive data, and we do not identify any significant ethical concerns arising from the methodology or experiments presented in this paper. All experiments were conducted using publicly available datasets and models in accordance with their respective licenses and terms of use.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the proposed method, training procedure, datasets, evaluation protocols, and hyperparameters in the main paper and appendix. Additional implementation details and experimental settings are included in the supplementary materials. We will also release the source code and configurations to facilitate reproduction of the reported results.

REFERENCES

[1] Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. Retaining by doing: The role of on-policy data in mitigating forgetting. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2510.18874.

[2] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

[3] Guangya Hao, Yunbo Long, and Zhuokai Zhao. Self-evolving multi-agent systems via decentralized memory. *arXiv preprint arXiv:2605.22721*, 2026. URL https://arxiv.org/abs/2605.22721.

[4] Guangya Hao, Yitong Shang, Yunbo Long, Zhuokai Zhao, and Hanxue Liang. Self-policy distillation via capability-selective subspace projection. *arXiv preprint arXiv:2605.22675*, 2026. URL https://arxiv.org/abs/2605.22675.

[5] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

[6] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021. doi: 10.3390/app11146421.

[7] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 2567–2577. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1259. URL https://aclanthology.org/D19-1259/.

[8] Song Lai, Haohan Zhao, Rong Feng, Changyi Ma, Wenzhuo Liu, Hongbo Zhao, Xi Lin, Dong Yi, Min Xie, Qingfu Zhang, Hongbin Liu, Gaofeng Meng, and Fei Zhu. Reinforcement fine-tuning naturally mitigates forgetting in continual post-training. *arXiv preprint arXiv:2507.05386*, 2025. URL https://arxiv.org/abs/2507.05386.

[9] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023. URL https://arxiv.org/abs/2305.20050.

[10] Jia Liu, ChangYi He, YingQiao Lin, MingMin Yang, FeiYang Shen, and ShaoGuo Liu. ETTRL: Balancing exploration and exploitation in LLM test-time reinforcement learning via entropy mechanism. *arXiv preprint arXiv:2508.11356*, 2025. URL https://arxiv.org/abs/2508.11356.

[11] Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *IEEE Transactions on Audio, Speech and Language Processing*, 33:3776–3786, 2025. doi: 10.1109/TASLPRO.2025.3606231.

[12] Math-AI Team. American mathematics competitions 2023 (AMC23). Hugging Face dataset. URL https://huggingface.co/datasets/math-ai/amc23. Contains 2023 AMC 12A and AMC 12B problems; accessed 2026-09-23.

[13] Yang Meng, Zhenya Liu, Zhuokai Zhao, and Yuxin Chen. Rethinking transfer in continual learning: A replay-based realisation. *arXiv preprint arXiv:2607.15587*, 2026. URL https://arxiv.org/abs/2607.15587.

[14] Meta AI. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. Meta AI, September 2024. URL https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/.

[15] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022.

[16] Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pp. 248–260. PMLR, 2022. URL https://proceedings.mlr.press/v174/pal22a.html.

[17] Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y. Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter space noise for exploration. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ByBAl2eAZ.

[18] Xin Qiu, Yulu Gan, Conor F. Hayes, Qiyao Liang, Yinggan Xu, Roberto Dailey, Elliot Meyerson, Babak Hodjat, and Risto Miikkulainen. Evolution strategies at scale: LLM fine-tuning beyond reinforcement learning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

[19] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

[20] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017. URL https://arxiv.org/abs/1703.03864.

[21] Frank Sehnke, Christian Osendorfer, Thomas Rückstieß, Alex Graves, Jan Peters, and Jürgen Schmidhuber. Policy gradients with parameter-based exploration for control. In *Artificial Neural Networks – ICANN 2008*, volume 5163 of *Lecture Notes in Computer Science*, pp. 387–396. Springer, 2008.

[22] Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2601.19897.

[23] Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. RL's razor: Why online reinforcement learning forgets less. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=7HNRYT4V44.

[24] Aozhe Wang, Zhengxi Lu, Jianze Wang, Shangke Lv, Ying Liu, Weiming Lu, Jun Xiao, Yueting Zhuang, Hua Yang, Qianglong Chen, and Yongliang Shen. TTPO: Test-time policy optimization. *arXiv preprint arXiv:2608.27448*, 2026. URL https://arxiv.org/abs/2608.27448.

[25] Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 10658–10671. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-emnlp.715. URL https://aclanthology.org/2023.findings-emnlp.715/.

[26] Chengyue Wu, Yukang Gan, Yixiao Ge, Zeyu Lu, Jiahao Wang, Ye Feng, Ying Shan, and Ping Luo. LLaMA pro: Progressive LLaMA with block expansion. In *Proceedings of the 62nd*

Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 6518–6537. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.352. URL https://aclanthology.org/2024.acl-long.352/.

[27] Tongtong Wu, Linhao Luo, Yuan-Fang Li, Shirui Pan, Thuy-Trang Vu, and Gholamreza Haffari. Continual learning for large language models: A survey. *arXiv preprint arXiv:2402.01364*, 2024. URL https://arxiv.org/abs/2402.01364.

[28] Kunbin Xu, Xingzuo Li, Xuefeng Bai, and Kehai Chen. Hi-TTRL: Regulating consensus with hints for test-time reinforcement learning. *arXiv preprint arXiv:2608.03545*, 2026. URL https://arxiv.org/abs/2608.03545.

[29] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

[30] Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2025. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/math-ai/aime25. Accessed 2026-09-23.

[31] Chu Zhao, Enneng Yang, Yuting Liu, Jianzhe Zhao, and Guibing Guo. ECHO: Entropy-confidence hybrid optimization for test-time reinforcement learning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

[32] Junhao Zheng, Shengjie Qiu, Chengming Shi, and Qianli Ma. Towards lifelong learning of large language models: A survey. *ACM Computing Surveys*, 57(8):1–35, 2025. doi: 10.1145/3716629.

[33] Yuhang Zhou, Lizhu Zhang, Yifan Wu, Mingyi Wang, Bo Peng, Jiayi Liu, Xiangjun Fan, and Zhuokai Zhao. OmniOPD: Logit-free on-policy distillation via speculative verification. *arXiv preprint arXiv:2606.01476*, 2026. URL https://arxiv.org/abs/2606.01476.

[34] Yuhang Zhou, Lizhu Zhang, Yifan Wu, Mingyi Wang, Bo Peng, Jiayi Liu, Xiangjun Fan, and Zhuokai Zhao. SAGE-OPD: Selective agent-guided intervention for multi-turn on-policy distillation. *arXiv preprint arXiv:2606.19659*, 2026. URL https://arxiv.org/abs/2606.19659.

[35] Yujun Zhou, Zhenwen Liang, Haolin Liu, Wenhao Yu, Kishan Panaganti, Linfeng Song, Dian Yu, Xiangliang Zhang, Haitao Mi, and Dong Yu. Evolving language models without labels: Majority drives selection, novelty promotes variation. *arXiv preprint arXiv:2509.15194*, 2025. URL https://arxiv.org/abs/2509.15194.

[36] Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, Biqing Qi, Youbang Sun, Zhiyuan Ma, Lifan Yuan, Ning Ding, and Bowen Zhou. TTRL: Test-time reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 38, pp. 131459–131483, 2025. doi: 10.52202/085713-4376. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/be690ea16f005c174f6c4102a5970e67-Abstract-Conference.html.

A Experimental Setup

A.1 Experiment Details

In this section, we provide detailed descriptions of our baselines implemented in our experiment.

TTRL. TTRL [36] performs reinforcement learning directly on unlabeled test-time data. For each prompt, it samples multiple responses from the current policy and uses majority voting over the extracted answers to obtain a pseudo-label. Rollouts that agree with the pseudo-label receive positive rewards, which are then used for policy optimization.

ECHO. ECHO [31] extends test-time reinforcement learning by improving both rollout exploration and policy optimization. It combines entropy–confidence guided rollout generation with confidence-aware advantage shaping to reduce early pseudo-label bias and rollout collapse.

EVOL-RL. EVOL-RL [35] augments majority-based test-time reinforcement learning with an explicit novelty signal. In addition to rewarding agreement with the majority answer, it encourages semantically distinct trajectories to maintain reasoning diversity and reduce entropy collapse.

ES. ES [18] performs zeroth-order optimization directly in parameter space. It samples a population of perturbed policies, evaluates each member using a scalar fitness value, and aggregates their fitness-weighted perturbations to update the model. We adapt ES to our test-time setting by applying it to the same unlabeled prompts and using the corresponding self-generated reward signal for population evaluation.

SFT. Supervised fine-tuning (SFT) [15] directly increases the likelihood of target responses using a token-level cross-entropy objective. Since ground-truth answers are unavailable in our setting, we adapt SFT with self-generated pseudo-targets from the test-time data. The resulting baseline therefore uses the same unlabeled inputs as ESRL.

A.2 Evaluation Metrics

Let Q denote the number of evaluation questions and n the number of sampled responses per question. For each question x_i, let $\mathcal{Y}_i = \{y_{i,j}\}_{j=1}^{n}$ denote its n sampled responses, $A(y)$ the normalized final answer extracted from response y, and a_i^* the ground-truth answer.

We consider two evaluation modes. In the *center-policy* mode, all n responses are sampled from the current student policy π_θ. In the *population-policy* mode, responses are pooled across the ES-perturbed policies. Unless otherwise stated, we use $n = 32$, corresponding to 32 responses from the center policy or two responses from each of the $N = 16$ perturbed policies.

Pass@n. Pass@n measures whether at least one of the n sampled responses produces the correct final answer:

$$\mathrm{Pass@}\, n = \frac{1}{Q}\sum_{i=1}^{Q} \mathbf{1}\left[\exists\, j \in \{1,\dots,n\} : A(y_{i,j}) = a_i^*\right].$$

We report Pass@1 and Pass@32. Pass@1 measures single-sample performance, while Pass@32 measures whether correct solutions remain accessible under a larger sampling budget, providing a measure of reasoning coverage.

Majority@n. For each question, we select the most frequent answer among the n sampled responses:

$$\hat{a}_i = \arg\max_{a} \sum_{j=1}^{n} \mathbf{1}\left[A(y_{i,j}) = a\right],$$

and compute

$$\mathrm{Maj@}\, n = \frac{1}{Q}\sum_{i=1}^{Q} \mathbf{1}\left[\hat{a}_i = a_i^*\right].$$

Under center-policy evaluation, Majority@n measures whether repeated sampling from the student forms a correct consensus. Under population-policy evaluation, it measures whether the perturbed-policy population collectively forms a correct consensus. Unparseable responses abstain from voting, and ties are resolved by the first observed tied answer.

Answer Entropy. For question i, let $n_{i,a}$ denote the number of parseable responses among the n samples whose normalized answer is a. We define

$$p_i(a) = \frac{n_{i,a}}{\sum_{a'} n_{i,a'}},$$

and compute

$$H_i = -\sum_a p_i(a) \log p_i(a), \qquad \overline{H} = \frac{1}{Q} \sum_{i=1}^{Q} H_i.$$

Unparseable responses are excluded from the entropy calculation. Under center-policy evaluation, answer entropy captures stochastic diversity within a single policy. Under population-policy evaluation, it additionally captures diversity across different perturbed policies introduced by parameter-space exploration.

Mean Response Length. Let $\ell(y)$ denote the number of generated completion tokens in response y. We compute

$$\overline{L} = \frac{1}{nQ} \sum_{i=1}^{Q} \sum_{j=1}^{n} \ell(y_{i,j}).$$

We use this metric to track changes in reasoning length during continual adaptation.

A.3 Implementation Details

For hyperparameters, we use the AdamW optimizer with `bfloat16` precision and apply gradient clipping with a maximum norm of 1.0. For rollout generation, we use a default group/population size of 16, with each population member generating one response. Responses are sampled with a temperature of 0.8 and top-p of 0.95. For optimization, we use a learning rate of 1×10^{-5} for TTRL, ECHO, EVOL-RL, and ESRL, while SFT uses 5×10^{-5} and ES uses a step size of 5×10^{-4}. The KL coefficient is set to 1×10^{-3}, and the PPO clipping parameter is 0.2. For ESRL, we set the perturbation standard deviation to $\sigma = 1 \times 10^{-3}$ and refresh the anchor every 10 prompts. Unless otherwise specified, the batch size is set to 2 for all methods, and $p_{\max} = 0.95$. We set the number of episodes to 5, 4, and 4 for MATH-500, MMLU, and MedQA, respectively, according to the dataset size. The maximum generation length is set to 8,192 tokens for all methods. All experiments are conducted on 8× NVIDIA H200 141 GB GPUs.

B Additional Experiment and Ablations results

B.1 Multi-Sample Evaluations

Table 5 presents the Pass@32 results for source-benchmark evaluation and in-domain generalization. The results are consistent with the main findings: ESRL demonstrates strong generalization, achieving the best average performance and ranking first or second on 8 out of 9 benchmarks.

Table 5: Pass@32 results for source-benchmark evaluation and in-domain generalization using Qwen3-0.6B.

	Math			General Knowledge			Medical			
Method	**MATH-500**	**AMC23**	**AIME25**	**MMLU**	**ARC-C**	**GPQA-D**	**MedQA**	**MedMCQA**	**PubMedQA**	**Avg.**
Base model	**45.0**	33.0	6.7	73.0	85.0	**63.9**	81.0	83.0	**90.7**	62.4
TTRL	31.0	36.1	6.7	75.0	**92.0**	54.0	**90.0**	73.0	90.0	60.9
EVOL-RL	4.0	**41.0**	0.0	**86.0**	**92.0**	59.1	86.0	68.0	84.0	57.8
ECHO	43.0	37.3	3.6	62.0	79.0	34.8	88.0	72.0	87.0	56.3
ESRL	**45.0**	38.6	**13.3**	82.0	**92.0**	57.4	88.0	**88.0**	90.0	**66.0**

B.2 Effect of Noise Scale on Performance

Table 6 reports the Pass@1 performance on the general knowledge domain under different values of σ. The default setting, $\sigma = 0.001$, achieves the best performance on two of the three benchmarks and the highest average score. These results also show that ESRL remains robust across different choices of σ. Notably, setting $\sigma = 0$, corresponding to ESRL w/o Parameter Perturbation, removes parameter-space exploration while keeping the remaining ESRL pipeline unchanged, reducing the average accuracy from 50.3 to 46.2 and confirming the benefit of parameter-space exploration.

B.3 Computational Cost Analysis

We compare the computational cost of ESRL and TTRL using Qwen3-0.6B on the Math domain under the same experimental configuration on NVIDIA H200 GPUs. Each method is independently repeated three times, and we report the median runtime together with throughput, peak GPU memory, average GPU utilization, and energy consumption in Table 7. Compared with TTRL, ESRL

Table 6: Pass@1 results under different perturbation scale σ using Qwen3-0.6B.

σ	General Knowledge			Avg.
	MMLU	ARC-C	GPQA-D	
0.0	45.5	68.5	24.7	46.2
0.0005	47.0	69.5	27.8	48.1
0.001	**49.5**	**70.5**	30.8	**50.3**
0.005	46.5	**70.5**	**31.3**	49.4
0.01	48.5	**70.5**	27.3	48.8

incurs a 14.0% runtime overhead and a 12.3% reduction in throughput. This additional cost mainly arises from evaluating multiple parameter-perturbed policies during population exploration. ESRL also shows 8.6 percentage points higher average GPU utilization and 24.0% higher energy consumption. In contrast, peak GPU memory changes by less than 0.2%. This is because ESRL does not store all perturbed models in memory at once. Instead, each population member is represented by a random seed, and its perturbation is regenerated when needed through our memory-efficient perturbation implementation. As a result, only the anchor parameters and one temporary perturbation buffer are kept in memory at any time. Overall, ESRL adds moderate computational overhead while keeping peak memory usage nearly unchanged.

Table 7: Runtime and computational cost comparison between ESRL and TTRL using Qwen3-0.6B on the Math domain. Results are measured over three independent runs.

Method	Runtime (min)	Prompts/h	Peak Mem. (GiB)	GPU Util.	Energy (Wh)
TTRL	78.85	304.38	100.94	67.6%	303.3
ESRL	89.90	266.96	100.78	76.2%	376.0

B.4 Robustness Across Random Seeds

Table 8 reports the performance of each method over three seeds. The results show that all methods exhibit relatively low variance and stable performance across runs.

Table 8: Pass@1 performance of different methods on the General Knowledge domain, reported as mean $\pm$ standard deviation over three seeds using Qwen3-0.6B.

Method	General Knowledge			Avg.
	MMLU	ARC-C	GPQA-D	
Base model	$44.0_{\pm 0.0}$	$65.5_{\pm 0.0}$	$26.3_{\pm 0.3}$	45.3
TTRL	$44.5_{\pm 1.5}$	$70.0_{\pm 1.6}$	$25.8_{\pm 2.8}$	46.8
ECHO	$46.0_{\pm 1.0}$	$69.5_{\pm 1.0}$	$23.7_{\pm 1.1}$	46.4
Evol-RL	$46.5_{\pm 2.3}$	$69.5_{\pm 0.3}$	$27.3_{\pm 1.3}$	<u>47.8</u>
ESRL	$49.5_{\pm 1.3}$	$70.5_{\pm 0.3}$	$30.8_{\pm 1.3}$	**50.3**

C Failure Mode Discussion

Hyperparameter settings play an important role in RL training, and their effects are further amplified in continual test-time learning, where supervision is derived entirely from self-generated signals. As shown in Table 9, increasing the batch size to 128 results in only marginal gains over the base model. Unlike smaller batches, where sparse informative signals can have a strong influence on each update and drive rapid adaptation, an excessively large batch averages these signals with many weak or conflicting updates. This dilutes useful stochastic exploration and prevents newly discovered reasoning modes from being effectively reinforced. Consequently, excessively large batches substantially weaken the adaptation signal and lead to little improvement during continual test-time learning.

Table 9: Failed attempts with a large batch size using Qwen3-0.6B.

Setting	Math			Avg.
	MATH-500	AMC23	AIME25	
Base Model	5.0	**6.0**	**0.0**	3.7
Batch Size = 128	**6.0**	**6.0**	**0.0**	**4.0**

Algorithm 1 Evolution Strategy Reinforcement Learning (ESRL)

Require: Unlabeled prompts $\mathcal{D}$; student parameters θ; frozen reference policy π_{θ_0}; population size N; perturbation radius σ; block size B; agreement threshold $p_{\max}$; KL weight λ; learning rate η

for each block $\mathcal{B}$ of at most B prompts in $\mathcal{D}$ **do**
  $\theta_{\text{anc}} \leftarrow \theta$ ▷ Synchronise the anchor with the student
  Sample perturbation seeds $s_1, \ldots, s_N$
  for $k = 1$ to N **do**
    $\varepsilon_k \leftarrow \varepsilon(s_k)$
    $\theta_k \leftarrow \theta_{\text{anc}} + \sigma\varepsilon_k$
    $\{y_k^{(x)}\}_{x\in\mathcal{B}} \leftarrow \textsc{GenerateBatch}(\pi_{\theta_k}, \mathcal{B})$
  end for
  for each prompt $x \in \mathcal{B}$ **do**
    $\{a_k\}_{k=1}^N \leftarrow \{\text{Ans}(y_k^{(x)})\}_{k=1}^N$
    $a^\star, \{r_k\}_{k=1}^N \leftarrow \textsc{MajorityVote}(\{a_k\}_{k=1}^N)$ ▷ equation 3
    if $\textsc{ValidityGate}(\{a_k\}_{k=1}^N, p_{\max})$ is false **then**
      continue
    end if
    $\{A_k\}_{k=1}^N \leftarrow \textsc{Standardise}(\{r_k\}_{k=1}^N)$ ▷ equation 4
    $\theta \leftarrow \theta - \eta\nabla_\theta\mathcal{L}(\theta)$ ▷ equation 5
  end for
end for