# Efficient Reinforcement Learning for Large Language Models: A Survey

**Anonymous ACL submission**

## Abstract

Reinforcement learning (RL) remains a fundamental computational challenge in Large Language Model (LLM) post-training due to the inherent complexity of the RL paradigm. With the advent of complex reasoning and agentic tasks, addressing these efficiency bottlenecks has become even more critical. In this paper, we explore the structural inefficiencies of the RL pipeline and the implications of physical acceleration. We define the root bottlenecks in expensive rollout generation, sparse training signals, and low throughput in synchronous systems (RQ1), and review current acceleration methods across the training pipeline, including data selection, speculative decoding, and asynchronous system decoupling (RQ2). Crucially, we investigate the efficiency-stability trade-off induced by these physical accelerations (RQ3) through an empirical case study that demonstrates how policy staleness paralyzes standard trust-region mechanisms. Grounded in this pathology, we synthesize related work to enable more stable RL training under asynchronous training and inference. Finally, we identify open problems and future research directions, especially in asynchronous agentic settings, offering a comprehensive review to guide scalable RL system design.

## 1 Introduction

Large language models (LLMs) have demonstrated exceptional capabilities in complex reasoning and alignment, driven largely by reinforcement learning (RL) paradigms such as RLHF and RLVR (Achiam et al., 2023; Guo et al., 2025; Chen et al., 2025). However, scaling RL introduces profound computational and structural challenges. Unlike supervised fine-tuning (SFT), which efficiently optimizes over static datasets, RL relies on on-policy rollouts (Chen et al., 2025; Achiam et al., 2023; Guo et al., 2025). The model must repeatedly generate trajectories to explore the environment, compute rewards, and estimate policy gradients. Due to the sequential nature of autoregressive decoding, the cost of rollout generation scales linearly with sequence length and model size, frequently dominating the wall-clock time and computational budget of the entire post-training pipeline (Chen et al., 2025; Sheng et al., 2025).

Beyond the sheer computational cost of sampling, efficient RL for LLMs is further bottlenecked by structural inefficiencies across the training pipeline. At the data level, a massive volume of generated rollouts often yields sparse or zero-advantage signals, wasting compute on uninformative updates (Guo et al., 2025; Yu et al., 2026). At the system level, traditional synchronous RL frameworks tightly couple inference (rollout generation) with training (policy updates), resulting in severe hardware underutilization and limited throughput (Sheng et al., 2025). To mitigate these bottlenecks, a rapidly growing body of literature has proposed diverse acceleration interventions—ranging from difficulty-aware data filtering and speculative decoding to fully asynchronous architectures.

Yet, these emerging efforts remain highly fragmented, obscuring a holistic understanding of how data, sampling, and system-level efficiencies interact. Furthermore, pursuing high throughput via aggressive optimizations—such as trajectory reuse, low-precision sampling, or asynchronous execution—inherently breaks the on-policy assumption (Zheng et al., 2025; Lei et al., 2026; Shen et al., 2026). This induces severe distribution shifts, policy staleness, and gradient bias, creating a critical gap in understanding how to systematically balance computational efficiency with stable policy convergence.

In this paper, we present the first comprehensive survey dedicated specifically to the efficiency of reinforcement learning for LLMs. While existing surveys broadly cover LLM reasoning (Zhang et al., 2025b), general RL paradigms (Srivastava and Aggarwal, 2025), or specialized applications (Zhang

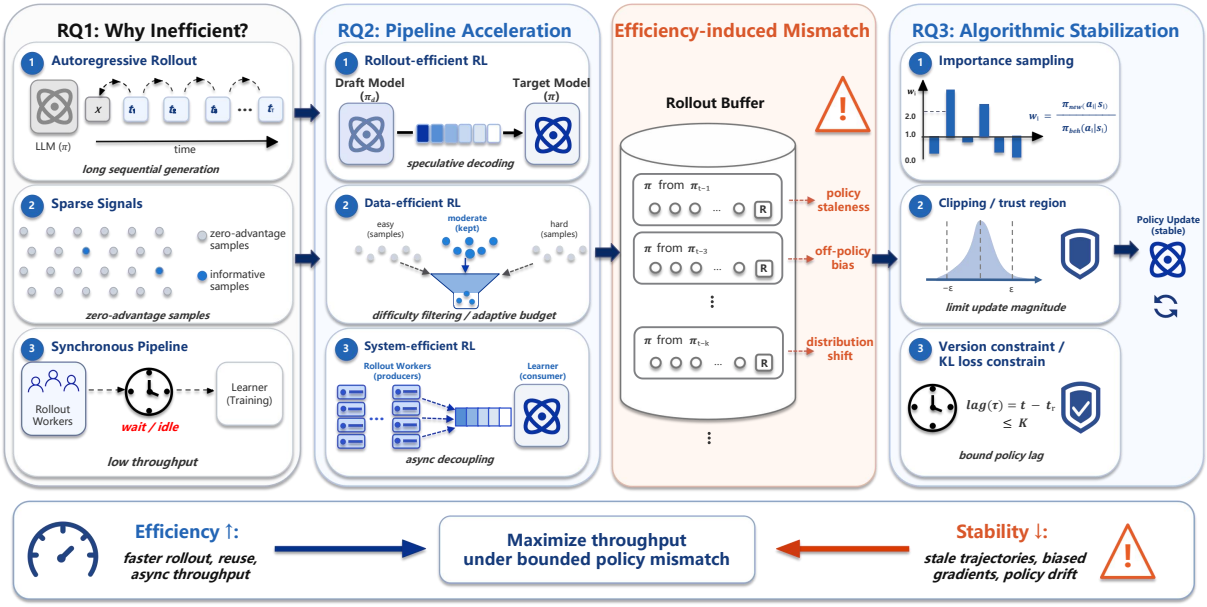



Figure 1: Overview of our survey. RL inefficiency stems from costly rollout generation, sparse effective training signals, and low-throughput pipelines (RQ1). Existing methods improve RL efficiency via rollout acceleration, data selection, and training- inference decoupling (RQ2). These accelerations, however, induce policy staleness and off-policy bias, requiring algorithmic stabilization (RQ3). The key challenge is to make the trade-off between efficiency and stability.

et al., 2026a), they typically treat training efficiency as a peripheral engineering concern rather than a core challenge. By structuring the fragmented literature through the lens of the complete training pipeline, we clarify the underlying mechanisms of current acceleration techniques. Crucially, we move beyond merely cataloging these methods to deeply analyze the fundamental trade-offs they introduce, demonstrating why high-throughput pipelines implicitly demand advanced algorithmic stabilization.

**Organization of this Survey.** As shown in Figure 1, we navigate this field through three core research questions. Section 2 addresses RQ1 (Why is RL for LLMs inefficient?) by dissecting the structural bottlenecks across rollout generation, data utilization, and synchronous system execution. Section 3 tackles RQ2 (How can the RL pipeline be made more efficient?) by categorizing state-of-the-art methods into data density improvement, rollout acceleration, and system-level decoupling. Section 4 explores RQ3 (The trade-off between efficiency and stability.) by introducing an empirical case study on how policy staleness halts standard RL convergence, followed by a systematic review of algorithmic stabilization strategies, such as off-policy correction and trust-region constraints. Finally, Section 5 outlines open challenges and future directions for algorithm-system co-design.

## 2 Why is RL for LLMs Inefficient? (RQ1)

A standard RL optimization step for LLMs spans a complex lifecycle of trajectory sampling, reward evaluation, and policy updates (Ouyang et al., 2022). Diagnosing the efficiency losses accumulated throughout this lifecycle mandates a granular, structural analysis. In this section, we formulate a three-dimensional framework to categorize these pipeline inefficiencies, and we deeply analyze the massive computational overhead of rollout generation, the data inefficiency driven by sparse training signals, and the synchronization stalls inherent to standard decoupled RL systems.

### 2.1 Cost of Rollout Generation

The most immediate computational bottleneck in the RL pipeline emerges from trajectory generation. While autoregressive decoding is inherently expensive, the recent paradigm shift toward complex reasoning (Chen et al., 2025) and agentic capabilities has magnified this cost by orders of magnitude. Initially, training models for advanced mathematical and coding tasks via Chain-of-Thought (CoT) (Wei et al., 2022) naturally extended sequence lengths to thousands of tokens, linearly scaling the generation latency.

Furthermore, as reinforcement learning scales to Agentic RL, models are increasingly required to interact with external tools and dynamic environments (Wei et al., 2025; Li et al., 2025). This introduces severe I/O bottlenecks, as the rollout process is frequently halted to wait for environment feedback (e.g., executing Python code or web browsing). Most recently, the emergence of auto-research (Ma et al., 2026a) and autonomous agents tasks models with multi-step exploration spanning dozens of hours (Ma et al., 2026b; Ding et al., 2026). In these long-horizon scenarios, the sheer length of episodes and the necessity for repeated environmental interactions transform rollout generation from a standard inference step into a paralyzing systemic bottleneck, rendering traditional on-policy synchronization computationally intractable.

### 2.2 Sparse Signals in Generated Data

Modern RL pipelines are fundamentally limited by a severe data-efficiency bottleneck: zero-gradient rollouts severely restrict learning. This bottleneck is especially pronounced in state-of-the-art critic-free paradigms such as Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which have become the standard for training reasoning LLMs. For a given prompt $x$, these methods sample $G$ trajectories $\{y_1, \ldots, y_N\}$ and compute group-relative advantages:

$$A_i = \frac{R_i - \text{mean}(R_1, ...R_G)}{\text{std}(R_1, ...R_G)} \tag{1}$$

The mathematical structure ensures that prompts misaligned with model capabilities produce no learning signal. When prompts are too easy (policy succeeds on all $N$ samples) or too difficult (fails on all $N$), reward variance disappears ($R_i = \text{mean}(R_1, ...R_G)$), collapsing advantages $A_i$ to zero and nullifying policy gradients. This causes systemic inefficiency: a great computational effort is spent sampling $N$ reasoning trajectories, yet optimization yields no discriminative information.

### 2.3 Bottlenecks in Synchronous Systems

Traditional RL frameworks operate on a rigid synchronous loop, enforcing a strict synchronization barrier between rollout generation, reward computation, and policy updates (von Werra et al., 2020; Sheng et al., 2025). This tightly coupled architecture inherently introduces massive execution bubbles—periods where GPU compute is completely unutilized. This structural hardware underutilization manifests across two critical dimensions.

First, at the inter-phase level, inference and training exhibit fundamentally conflicting hardware profiles. Rollout generation is memory-bandwidth bound, whereas policy backpropagation is heavily compute-bound. In a synchronous graph, training hardware lies dormant during generation, and vice versa.

Second, at the intra-phase level, the system is severely paralyzed by the straggler effect. While the unpredictable sequence lengths of CoT reasoning already introduce significant batch latency, this bottleneck is catastrophically amplified in emerging Agentic RL scenarios. Because agents engage in highly variable multi-step interactions and tool executions, episode lengths become extremely skewed. The latency of an entire rollout batch is strictly dictated by its longest and most complex trajectory ($\max(T_i)$). Consequently, GPUs processing shorter, early-terminating episodes idle continuously, waiting for the straggler to complete (Fu et al., 2026). Together, these synchronization stalls impose a hard upper bound on pipeline throughput.

## 3 How Can RL for LLMs Be Made More Efficient? (RQ2)

To overcome the profound computational barriers of the LLM reinforcement learning pipeline, recent literature has developed a spectrum of acceleration techniques. Rather than viewing these optimizations as isolated tricks, we organize them into a unified taxonomy based on their target bottleneck. Specifically, this section categorizes current efficient RL methodologies into three distinct dimensions: (1) Sampling Acceleration (Section 3.1), which directly mitigates the latency of autoregressive rollout generation; (2) Data-Centric Efficiency (Section 3.2), which maximizes the learning signal extracted from generated trajectories; and (3) Asynchronous Systems (Section 3.3), which decouple inference and training to unlock hardware throughput.

### 3.1 Accelerating Rollout Generation

Overcoming the fundamental latency of rollout generation requires circumventing its strict sequential dependency. The computational cost of rollout generation scales as $\mathcal{O}(N \cdot T \cdot |\theta|)$, dictated by the number of samples per prompt $N$, the sequence length $T$, and the model capacity $|\theta|$. Consequently,

state-of-the-art acceleration strategies operate by explicitly decoupling or minimizing $T$ and $|\theta|$ from the generation loop. We synthesize these methodologies into three primary paradigms:

**Speculative Decoding and Historical Drafting:** Speculative decoding directly attacks the sequential bottleneck by transforming memory-bound generation into parallelized, compute-bound verification. However, rather than training distinct draft models, recent frameworks exploit the high behavioral similarity across adjacent RL training epochs. Methods like SPEC-RL and SRT completely bypass model-based drafting by recycling previous iterations: SPEC-RL reuses historical trajectory segments as speculative prefixes (Liu et al., 2025a), while SRT constructs a per-prompt tree-structured cache of past continuations to serve as a model-free draft (Chang et al., 2026). Expanding on this temporal redundancy, RhymeRL introduces HistoSpec to draft from historical tokens and pairs it with HistoPipe, a scheduling strategy tailored to historical rollout-length distributions (He et al., 2025). To combat intra-phase stragglers, SPECACTOR introduces a decoupled speculative engine utilizing Fastest-of-N selection, specifically explicitly targeting the long-tail rollout bottleneck (Cheng et al., 2025). Ultimately, when integrated into massive asynchronous backends, this speculation acts as a necessary acceleration primitive—slashing rollout latency by 2–3$\times$ while preserving exact on-policy target distributions (Iso et al., 2026).

**Precision Scaling and Execution Co-Design:** To bypass the $\mathcal{O}(|\theta|)$ memory-bandwidth bottleneck during rollout generation, recent frameworks aggressively scale down the rollout policy's precision. Frameworks like FP8-RL deploy W8A8 linear layers and KV-cache quantization, pairing physical speedups with strict importance-sampling corrections (Qiu et al., 2026). Similarly, QuRL maintains a full-precision trainer but executes rollouts via a quantized actor, neutralizing off-policy drift through update-aware quantization and adaptive clipping ranges (Li et al., 2026b). At the frontier scale, DeepSeek-V4 pushes compression to extreme FP4 precision by integrating Quantization-Aware Training (QAT) into the RL pipeline. By natively targeting MoE expert weights and long-context attention (QK) paths in FP4, it structurally aligns rollout behavior with deployment while halving top-$k$ routing memory (DeepSeek-AI, 2026). Furthermore, precision scaling is frequently synergized with decoding optimizations. GLM-5 pairs FP8 rollouts with Multi-Token Prediction (MTP), a mechanism exceptionally effective under small-batch regimes, to explicitly neutralize the long-tail latency of complex reasoning stragglers (Zeng et al., 2026).

**Trajectory Slicing and Prefix Sharing:** Rather than accelerating token-level decoding, trajectory-level optimizations dynamically manage the effective sequence length $T$ to eliminate computational redundancy and long-tail stragglers. To mitigate redundant prompt processing across $N$ samples, AREAL-DTA structures rollouts as prefix trees; it employs a DFS-based dynamic tree attention to inherently share prefix computations during both forward generation and backward optimization passes (Zhang et al., 2026b). To combat the severe synchronization stalls caused by extreme-length trajectories, APRIL introduces active partial rollouts. By over-provisioning requests and preemptively stopping once sufficient responses complete, it recycles incomplete stragglers for continuation in subsequent steps (Zhou et al., 2025). Similarly targeting the long-tail latency bottleneck, Seer implements divided rollouts paired with context-aware scheduling to balance sequence workloads across synchronous pipelines (Qin et al., 2025). By intelligently sharing prefixes and slicing the generation horizon, these methods actively constrain the upper bound of $T$, preserving massive GPU cycles.

### 3.2 Improving Data Efficiency

To mitigate the zero-variance pathology inherent in critic-free algorithms (e.g., GRPO), generation budgets must be exclusively routed toward prompts that yield high gradient information. Rather than treating prompt selection as parallel techniques, recent literature exhibits a clear evolutionary trajectory toward maximizing information density:

**From Reactive Curricula to Proactive Difficulty Prediction:** To maximize gradient signal-to-noise ratio (SNR), data selection must target the Zone of Proximal Development—prompts of intermediate difficulty. Early strategies relied on reactive curricula and heuristic filtering. For instance, Ji et al. (2025) employ staged exposure to progressively harder mathematical tasks, while Bae et al. (2026) explicitly filter batches around intermediate pass rates to maximize useful signals. To automate this, frameworks like ADARFT dynamically calibrate target difficulty thresholds based on recent

reward feedback (Shi et al., 2025), and VCRL utilizes rollout reward variance as a direct proxy to isolate high-signal samples (Jiang et al., 2025b).

However, evaluating exact difficulty via exhaustive rollouts is computationally prohibitive. Consequently, the paradigm shifted toward efficient proxy estimations and proactive prediction. To bypass repeated rollouts, PCL trains a dedicated value model (Gao et al., 2025), and UFO-RL leverages single-pass uncertainty estimates to filter incoming data (Zhao et al., 2026). Pushing this further, SPEED-RL validates the SNR benefits of intermediate prompts while proposing a highly efficient online estimation curriculum (Zhang et al., 2025c). At the frontier of this evolution, MoPPS and BOTS formulate difficulty as a latent variable, employing Bayesian inference, posterior sampling, and Thompson sampling to proactively predict prompt success rates before any generation occurs (Qu et al., 2026; Shen et al., 2025). Transcending static datasets entirely, RLVE introduces procedural verifiable environments that adaptively scale problem complexity to match the evolving policy (Zeng et al., 2025).

**Selective Rollouts and Dynamic Budgeting:** Advancing beyond basic difficulty filtering, recent architectures optimize efficiency at the trajectory generation granularity. The foundational approach to mitigating zero-advantage groups relies on retroactive filtering. For instance, DAPO implements a dynamic sampling policy that drops zero-gradient samples after generation and triggers resampling to maintain useful batches (Yu et al., 2026). Similarly, PODS executes maximum-variance down-sampling, generating massive rollouts but restricting the memory-heavy policy updates exclusively to the most informative subset (Xu et al., 2025). However, while these retroactive methods secure stable gradients, their efficiency gains are sub-optimal, as the immense forward-pass compute required to generate the uninformative rollouts has already been irreversibly incurred.

To reclaim this wasted generation compute, modern frameworks shift toward proactive budgeting. Intervening mid-generation, AERO utilizes Bayesian posterior estimates to aggressively prune uninformative trajectories early, avoiding zero-advantage dead zones (Zhang et al., 2026c). Pushing optimization to the pre-rollout phase, methods actively break the uniform $\mathcal{O}(N)$ sampling budget. GRESO employs a lightweight pre-filtering algorithm to predict and completely skip uninformative prompts based on temporal reward consistency (Zheng et al., 2026b). Concurrently, VIP frames budget allocation as a convex optimization problem, employing Gaussian processes to dynamically assign rollout counts proportional to expected gradient variance (Nguyen et al., 2026). Further amplifying these savings, Sun et al. (2026) combine target selection (focusing on success probabilities near 0.5) with a FIFO rollout replay buffer, systematically minimizing both the number of training steps and the per-step generation burden.

**Data Reactivation and Compositional Synthesis:** Orthogonal to filtering, techniques have emerged to extract value from discarded data. ERPO induces generation diversity via temperature scaling to reactivate gradients from solved prompts (Liu et al., 2025b), while Composition-RL synthesizes simple prompts into complex queries to expand the dataset boundary (Xu et al., 2026). Critically, however, this aggressive pursuit of high-variance signals via continuous filtering and reuse inherently warps the training distribution. This breaks the fundamental assumption of on-policy sampling, directly introducing off-policy bias that necessitates advanced algorithmic stabilization.

### 3.3 System-Level Decoupling and Orchestration

To eliminate the massive execution bubbles intrinsic to synchronous RL, modern architectures fundamentally dismantle the tightly coupled training loop. This system-level evolution specifically targets hardware starvation across inter-phase and intra-phase dimensions, culminating in flexible abstractions for agentic workflows:

**Eliminating Inter-Phase Bubbles via Asynchronous Decoupling:** The foundational step in throughput optimization is breaking the global synchronization barrier between rollout generation and policy updates. Frameworks like AsyncFlow (Han et al., 2025) replace rigid stage-by-stage execution with a streaming producer-consumer dataflow, allowing trainers to continuously consume data while generation proceeds. To scale this asynchrony to massive models, LlamaRL (Wu et al., 2025) introduces GPU-native distributed weight synchronization via NVLink, completely offloading generation and sustaining peak utilization even at the 405B scale. Furthermore, systems like ECHO-2 (Xiao et al., 2026) and AstraFlow (Zheng et al.,

2026a) treat weight dissemination and data staleness as explicit system provisioning variables, using pipelined broadcast and pull-based sparse updates to absorb wide-area communication latency.

**Mitigating Intra-Phase Bubbles via Trajectory Scheduling:** Beyond phase decoupling, systems must conquer the extreme intra-batch straggler effect caused by variable reasoning lengths. To tame this skewness within synchronous boundaries, Seer (Qin et al., 2025) employs online context-aware scheduling to pack and balance heterogeneous requests. In fully asynchronous settings, AReaL (Fu et al., 2026) directly eliminates intra-phase dormancy through interruptible rollout workers: it aggressively truncates overly long trajectories, commits the partial data for immediate training, and seamlessly resumes generation under updated weights. By jointly optimizing trajectory staleness and length skewness, frameworks like StaleFlow (Li et al., 2026a) ensure that long-tail stragglers no longer paralyze global cluster throughput.

**Framework Abstractions for Agentic RL:** As optimization targets shift toward complex multi-turn reasoning and tool-augmented agents, traditional monolithic RL scripts suffer from severe CPU-GPU synchronization bubbles. To resolve this, recent architectures redesign Agentic RL around asynchronous, service-oriented paradigms. Specifically, *slime* (Zhu et al., 2025) abstracts generation into Server-Based Rollouts, mirroring standard API services to seamlessly integrate external agent frameworks, while utilizing an intelligent Rollout Buffer to asynchronously filter and process messy message-list trajectories. To tackle the massive latency of environment interactions, VERLTOOL(Jiang et al., 2025a) decouples action generation from execution via a unified Tool Server. By enabling trajectory-by-trajectory asynchronous rollouts, it explicitly overlaps LLM inference with external API executions (e.g., Python, web search), eliminating hardware idle bubbles . Pushing architectural modularity further, AstraFlow (Zheng et al., 2026a) introduces a programmable dataflow layer and Rollout-as-a-Service (RaaS), providing the essential infrastructure for multi-policy collaborative training and heterogeneous backend orchestration.

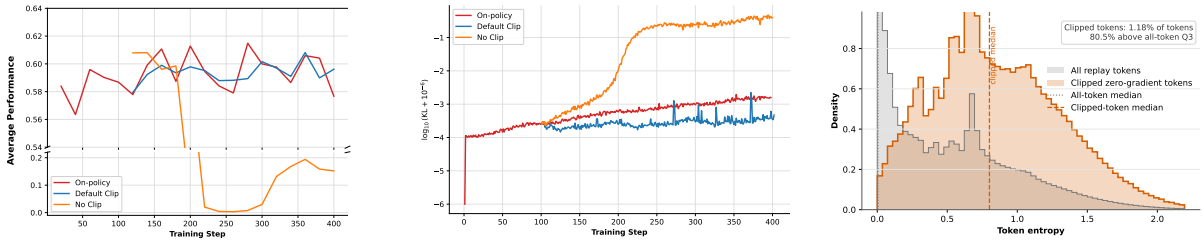
# 4 Trade-off between Efficiency and Stability (RQ3).

The pursuit of system-level throughput and data efficiency is not without cost. Modern efficient RL pipelines fundamentally violate the core assumptions of on-policy optimization by training on trajectories generated by outdated policy versions or perturbed inference engines. This temporal and semantic mismatch, formally identified as Policy Staleness and Off-Policy Bias, creates an inherent efficiency-stability trade-off. This section analyzes why this phenomenon destabilizes standard RL and synthesizes the algorithmic strategies developed to reclaim performance.

## 4.1 Empirical Case Study: The Failure of Standard Trust Regions

Standard on-policy RL algorithms (e.g., PPO, GRPO) rigidly assume that the training data distribution strictly matches the current policy ($\pi_\theta = \pi_{old}$). Systemic efficiency interventions—such as asynchronous decoupling or replay buffers—inherently violate this identity. To empirically diagnose how this staleness destabilizes optimization, we conduct a controlled replay experiment on Qwen-3-1.7B. We first train the model strictly on-policy, caching the generated trajectories and model checkpoints. We then resume training from an intermediate checkpoint using these stale trajectories via standard GRPO with clipping, alongside a no-clip variant (see Appendix A for details).

The results expose a profound algorithmic dilemma (Figure 2). Under standard clipping, performance rapidly stagnates (Figure 2a). Analysis of the KL divergence reveals a counter-intuitive pathology: the KL loss during standard replay is remarkably lower than during on-policy training (Figure 2b). This indicates that the trust-region constraint becomes excessively restrictive under staleness. Crucially, as shown in the entropy distribution of clipped tokens (Figure 2c), the clipping mechanism disproportionately penalizes high-entropy tokens. As corroborated by Zheng et al. (2025) and Wang et al. (2025), these high-entropy tokens predominantly correspond to critical CoT reasoning steps. By aggressively clipping these tokens, the optimizer becomes blind to the most vital learning signals. Conversely, completely removing the clip constraint (No-Clip) triggers catastrophic policy collapse, accompanied by violently exploding KL




(a) Performance (b) KL loss (c) Token entropy

Figure 2: Case study on the efficiency–stability trade-off. We compare training performance, KL loss, and the entropy distributions of all tokens and clipped tokens. The results illustrate that aggressive efficiency-oriented training can introduce policy staleness, destabilize trust-region optimization, and disproportionately affect high-entropy tokens. For detailed training setting, please refer to Appendix A.

divergence (Figure 2b).

This case study demonstrates that standard algorithmic settings are incompatible with high-throughput asynchronous training: strict clipping chokes the learning signal of complex reasoning, while unconstrained optimization invites fatal off-policy drift.

### 4.2 Algorithmic Stabilization

To counteract the optimization pathology exposed in Section 4.1, algorithmic interventions must mathematically neutralize off-policy bias. Recent literature addresses this by redesigning the optimization objective across three fundamental dimensions: adapting the trust region, reshaping importance weights, and enforcing global distributional constraints.

**Adaptive Trust Regions and Acceptance Bounds:** Standard fixed-bound clipping fails under high staleness by violently suppressing high-entropy reasoning tokens. To repair this, BAPO dynamically adjusts clipping boundaries to rebalance positive and negative gradient contributions, preventing entropy collapse (Xi et al., 2025). For extreme asynchrony, M2PO abandons hard token-level clipping entirely, proposing a second-moment constraint on importance weights that masks only extreme outliers while preserving vital reasoning signals (Zheng et al., 2025). Similarly, frontier architectures like GLM-5 implement IcePop-style double-sided token masking over rollout log-probabilities, effectively bounding the surrogate loss without requiring historical policy checkpoints (Zeng et al., 2026).

**Importance Weight Normalization and Reshaping:** A critical source of instability is the multiplicative variance explosion caused by token-level Importance Sampling (IS). To mitigate this, algorithms fundamentally reweight the gradient estimator. MinPRO demonstrates that while prefix-level IS is theoretically optimal, it requires a non-cumulative minimum-prefix surrogate to practically suppress variance (Lei et al., 2026). Alternatively, VESPO derives a variational reshaping kernel that operates directly on sequence-level weights (Shen et al., 2026), while SNIB introduces a self-normalized IS estimator with a baseline to guarantee asymptotic stability in critic-free setups (Anonymous, 2025). In decentralized networks, GEPO applies group-level expectation weighting to smooth latency-induced variance (Zhang et al., 2025a).

**Global Constraints and Alternative Objectives:** Beyond token-level adjustments, robust training requires trajectory-level version control and global regularizers. To safely ingest stale buffers, ReMix enforces a KL-convex constraint, utilizing policy reincarnation to smoothly transition from off-policy exploration to on-policy convergence (Liang et al., 2026). Addressing semantic mismatch (Guan et al., 2026), MIS-PO treats the inference engine as a proposal distribution, applying binary Metropolis-independence-sampling filters to discard rollouts whose geometric mean ratio drifts beyond recovery (Huang et al., 2026). Finally, algorithms like TBA completely bypass the PPO formulation, adopting the inherently off-policy Trajectory Balance objective to seamlessly digest asynchronous replay buffers without IS-induced instability (Bartoldson et al., 2026).

### 4.3 Pareto Frontier of Efficiency and Stability

The interplay between system throughput and algorithmic stabilization reveals that efficiency and stability are not mutually exclusive, but rather form a Pareto frontier. The optimal training configuration depends on balancing the computational cost of generation against the statistical penalty of off-policy updates. Recent analyses of experience replay demonstrate that strict on-policy sampling is mathematically suboptimal under high generation costs; trading a controlled increase in staleness-induced variance for reduced inference compute can save substantial budgets without degrading accuracy (Arnal et al., 2026). However, simply expanding the replay buffer is insufficient; batch construction must remain difficulty-aware and constrained to prevent homogeneous, zero-advantage updates (Wan et al., 2026). Ultimately, maximizing RL efficiency requires treating staleness not as an error, but as a budgeted resource.

## 5 Open Challenges and Future Directions

While the methodologies reviewed in this survey demonstrate significant progress in mitigating the physical and algorithmic bottlenecks of LLM reinforcement learning, the transition from isolated efficiency techniques to holistic, scalable paradigms remains incomplete. Based on the field's current trajectory, we identify two critical open challenges that must guide future research.

### 5.1 The Need for Standardized Efficiency Benchmarks

A fundamental challenge in the current efficient RL literature is the severe fragmentation of evaluation metrics. Recent studies report acceleration using vastly different criteria—ranging from sample efficiency to system throughput . Furthermore, because different methods often achieve varying levels of final task performance, it is currently impossible to rigorously evaluate the true Pareto frontier between convergence speed and ultimate reasoning capability.

To address this, the field needs standardized efficiency benchmarks tailored to LLM RL. Future evaluations must require "Time-to-Accuracy" (the GPU-hours required to reach a baseline score) rather than throughput. Reporting system-level metrics, such as communication-to-compute overhead and bounded staleness, is essential. Only with these standards can researchers fairly compare asynchronous architectures to on-policy algorithms.

### 5.2 Deep Algorithm-System Co-design

As analyzed in Section 3 and 4, current research frequently treats system engineering and algorithmic stabilization as separate endeavors. Systems frameworks optimize for maximum throughput by forcefully decoupling generation and training, while algorithmic interventions attempt to correct the resulting off-policy bias retrospectively. This disjointed approach is reaching its limits; algorithms can no longer treat the underlying training framework as a black box.

The next leap in efficient RL necessitates deep algorithm-system co-design, where algorithms become topology- and hardware-aware. For instance, future RL optimizers could dynamically adjust their sampling targets and batch construction based on real-time KV-cache residency across a distributed GPU cluster, minimizing redundant memory offloading. Conversely, asynchronous system architectures should natively expose their latency bounds to the optimization graph, enabling the development of novel, staleness-resistant RL objectives that are intrinsically designed to operate under unpredictable hardware synchronization delays.

## 6 Conclusion

In this survey, we presented a comprehensive analysis of efficient reinforcement learning for large language models. By dissecting the RL pipeline, we identified the fundamental structural bottlenecks rooted in sampling, data sparsity, and synchronous execution (RQ1). We then synthesized the landscape of current strategies, mapping how difficulty-aware filtering, speculative generation, and asynchronous decoupled systems successfully maximize throughput (RQ2). Crucially, we highlighted that pursuing physical efficiency inherently violates on-policy assumptions, introducing a profound trade-off between system speed and optimization stability (RQ3). By unifying the literature on off-policy corrections and trust-region constraints, we demonstrated that true efficiency cannot be achieved solely through systems engineering. To truly advance scalable and agentic LLM training, we recommend establishing clear, standardized benchmarks for efficiency and prioritizing topology-aware algorithm-system co-design to integrate both optimization and engineering efforts.

## 7 Limitations

This survey has three primary limitations. (1) **Scope and Timeliness:** The LLM landscape evolves daily; hence, recent preprints may be omitted. Furthermore, we focus exclusively on text-based and agentic RL, omitting multimodal applications. (2) **Empirical Scale:** Due to computational constraints, our diagnostic case study evaluates a 1.7B model. Exhaustive benchmarking of asynchronous algorithms on frontier-scale models (70B+) remains unfeasible for typical academic settings. (3) **Industrial Opacity:** As frontier RL pipelines are highly guarded and closed-source, our analysis of system-level decoupling is heavily bounded by the current open-source ecosystem.

## References


Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anonymous. 2025. Principled policy optimization for LLMs via self-normalized importance sampling.

Charles Arnal, Vivien Cabannes, Taco Cohen, Julia Kempe, and Remi Munos. 2026. Efficient rl training for llms with experience replay. *arXiv preprint arXiv:2604.08706*.

Sanghwan Bae, Jiwoo Hong, Min Young Lee, Hanbyul Kim, JeongYeon Nam, and Donghyun Kwak. 2026. Online difficulty filtering for reasoning oriented reinforcement learning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 700–719.

Brian Bartoldson, Siddarth Venkatraman, James Diffenderfer, Moksh Jain, Tal Ben-Nun, Seanie Lee, Minsu Kim, Johan Obando Ceron, Yoshua Bengio, and Bhavya Kailkhura. 2026. Trajectory balance with asynchrony: Decoupling exploration and learning for fast, scalable llm post-training. *Advances in Neural Information Processing Systems*, 38:113901–113931.

Chi-Chih Chang, Siqi Zhu, Zhichen Zeng, Haibin Lin, Jiaxuan You, Mohamed S Abdelfattah, Ziheng Jiang, and Xuehai Qian. 2026. Srt: Accelerating reinforcement learning via speculative rollout with tree-structured cache. *arXiv preprint arXiv:2601.09083*.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*.

Rongxin Cheng, Kai Zhou, Xingda Wei, Siyuan Liu, Mingcong Han, Mingjing Ai, Yeju Zhou, Baoquan Zhong, Wencong Xiao, Rong Chen, and 1 others. 2025. Fast llm post-training via decoupled and fastest-of-n speculation. *arXiv preprint arXiv:2511.16193*.

DeepSeek-AI. 2026. Deepseek-v4: Towards highly efficient million-token context intelligence.

Shuangrui Ding, Xuanlang Dai, Long Xing, Shengyuan Ding, Ziyu Liu, Yang JingYi, Penghui Yang, Zhixiong Zhang, Xilin Wei, Xinyu Fang, and 1 others. 2026. Wildclawbench: A benchmark for real-world, long-horizon agent evaluation. *arXiv preprint arXiv:2605.10912*.

Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, and 1 others. 2026. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *Advances in Neural Information Processing Systems*, 38:36256–36282.

Zhaolin Gao, Joongwon Kim, Wen Sun, Thorsten Joachims, Sid Wang, Richard Yuanzhe Pang, and Liang Tan. 2025. Prompt curriculum learning for efficient llm post-training. *arXiv preprint arXiv:2510.01135*.

Zhong Guan, Yongjian Guo, Haoran Sun, Wen Huang, Shuai Di, Xiong Jun Wu, Likang Wu, and Hongke Zhao. 2026. Missing old logits in asynchronous agentic rl: Semantic mismatch and repair methods for off-policy correction. *arXiv preprint arXiv:2605.12070*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Zhenyu Han, Ansheng You, Haibo Wang, Kui Luo, Guang Yang, Wenqi Shi, Menglong Chen, Sicheng Zhang, Zeshun Lan, Chunshi Deng, and 1 others. 2025. Asyncflow: An asynchronous streaming rl framework for efficient llm post-training. *arXiv preprint arXiv:2507.01663*.

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

Jingkai He, Tianjian Li, Erhu Feng, Dong Du, Qian Liu, Tao Liu, Yubin Xia, and Haibo Chen. 2025. History rhymes: Accelerating llm reinforcement learning with rhymerl. *arXiv preprint arXiv:2508.18588*.

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

Ailin Huang, Ang Li, Aobo Kong, Bin Wang, Binxing Jiao, Bo Dong, Bojun Wang, Boyu Chen, Brian Li, Buyun Ma, and 1 others. 2026. Step 3.5 flash: Open frontier-level intelligence with 11b active parameters. *arXiv preprint arXiv:2602.10604*.

Hayate Iso, Tiyasa Mitra, Sudipta Mondal, Rasoul Shafipour, Venmugil Elango, Terry Kong, Yuki Huang, Seonjin Na, Izzy Putterman, Benjamin Chislett, and 1 others. 2026. Accelerating rl post-training rollouts via system-integrated speculative decoding. *arXiv preprint arXiv:2604.26779*.

Yunjie Ji, Sitong Zhao, Xiaoyu Tian, Haotian Wang, Shuaiting Chen, Yiping Peng, Han Zhao, and Xiangang Li. 2025. How difficulty-aware staged reinforcement learning enhances llms' reasoning capabilities: A preliminary experimental study. *arXiv preprint arXiv:2504.00829*.

Dongfu Jiang, Yi Lu, Zhuofeng Li, Zhiheng Lyu, Ping Nie, Haozhe Wang, Alex Su, Hui Chen, Kai Zou, Chao Du, and 1 others. 2025a. Verltool: Towards holistic agentic reinforcement learning with tool use. *arXiv preprint arXiv:2509.01055*.

Guochao Jiang, Wenfeng Feng, Guofeng Quan, Chuzhan Hao, Yuewei Zhang, Guohua Liu, and Hao Wang. 2025b. Vcrl: Variance-based curriculum reinforcement learning for large language models. *arXiv preprint arXiv:2509.19803*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Shiye Lei, Zhihao Cheng, and Dacheng Tao. 2026. A step back: Prefix importance ratio stabilizes policy optimization. *arXiv preprint arXiv:2601.22718*.

Haoyang Li, Sheng Lin, Fangcheng Fu, Yuming Zhou, Xiaodong Ji, Yanfeng Zhao, Lefeng Wang, Jie Jiang, and Bin Cui. 2026a. Unleashing efficient asynchronous rl post-training via staleness-constrained rollout coordination. *arXiv preprint arXiv:2601.12784*.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, and 1 others. 2025. Websailor: Navigating super-human reasoning for web agent. *arXiv preprint arXiv:2507.02592*.

Yuhang Li, Reena Elangovan, Xin Dong, Priyadarshini Panda, and Brucek Khailany. 2026b. Qurl: Efficient reinforcement learning with quantized rollout. *arXiv preprint arXiv:2602.13953*.

Jing Liang, Jinyi Liu, Yi Ma, Hongyao Tang, YAN ZHENG, Shuyue Hu, LEI BAI, and Jianye HAO. 2026. Squeeze the soaked sponge: Efficient off-policy RFT for large language model. In *The Fourteenth International Conference on Learning Representations*.

Bingshuai Liu, Ante Wang, Zijun Min, Liang Yao, Haibo Zhang, Yang Liu, Anxiang Zeng, and Jinsong Su. 2025a. Spec-rl: Accelerating on-policy reinforcement learning via speculative rollouts. *arXiv preprint arXiv:2509.23232*.

Chenxi Liu, Junjie Liang, Yuqi Jia, Bochuan Cao, Yang Bai, Heng Huang, and Xun Chen. 2025b. Explore data left behind in reinforcement learning for reasoning language models. *arXiv preprint arXiv:2511.04800*.

Tianyi Ma, Yiyue Qian, Zheyuan Zhang, Zehong Wang, Xiaoye Qian, Feifan Bai, Yifan Ding, Xuwei Luo, Shinan Zhang, Keerthiram Murugesan, and 1 others. 2026a. Autodata: A multi-agent system for open web data collection. *Advances in Neural Information Processing Systems*, 38:173416–173448.

Ziyu Ma, Shidong Yang, Yuxiang Ji, Xucong Wang, Yong Wang, Yiming Hu, Tongwen Huang, and Xiangxiang Chu. 2026b. Skillclaw: Let skills evolve collectively with agentic evolver. *arXiv preprint arXiv:2604.08377*.

Hieu Trung Nguyen, Bao Nguyen, Wenao Ma, Yuzhi Zhao, Ruifeng She, and Viet Anh Nguyen. 2026. Adaptive rollout allocation for online reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2602.01601*.

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

Ruoyu Qin, Weiran He, Weixiao Huang, Yangkun Zhang, Yikai Zhao, Bo Pang, Xinran Xu, Yingdi Shan, Yongwei Wu, and Mingxing Zhang. 2025. Seer: Online context learning for fast synchronous llm reinforcement learning. *arXiv preprint arXiv:2511.14617*.

Zhaopeng Qiu, Shuang Yu, Jingqi Zhang, Shuai Zhang, Xue Huang, Jingyi Yang, and Junjie Lai. 2026. Fp8-rl: A practical and stable low-precision stack for llm reinforcement learning. *arXiv preprint arXiv:2601.18150*.

Yun Qu, Qi Wang, Yixiu Mao, Vincent Tao Hu, Björn Ommer, and Xiangyang Ji. 2026. Can prompt difficulty be online predicted for accelerating rl finetuning of reasoning models? In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 1240–1250.

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

Guobin Shen, Chenxiao Zhao, Xiang Cheng, Lei Huang, and Xing Yu. 2026. Vespo: Variational sequence-level soft policy optimization for stable off-policy llm training. *arXiv preprint arXiv:2602.10693*.

Qianli Shen, Daoyuan Chen, Yilun Huang, Zhenqing Ling, Yaliang Li, Bolin Ding, and Jingren Zhou. 2025. Bots: A unified framework for bayesian online task selection in llm reinforcement finetuning. *arXiv preprint arXiv:2510.26374*.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, page 1279–1297. ACM.

Taiwei Shi, Yiyang Wu, Linxin Song, Tianyi Zhou, and Jieyu Zhao. 2025. Efficient reinforcement finetuning via adaptive curriculum learning. *arXiv preprint arXiv:2504.05520*.

Saksham Sahai Srivastava and Vaneet Aggarwal. 2025. A technical survey of reinforcement learning techniques for large language models. *Preprint*, arXiv:2507.04136.

Yifan Sun, Jingyan Shen, Yibin Wang, Tianyu Chen, Zhendong Wang, Mingyuan Zhou, and Huan Zhang. 2026. Improving data efficiency for llm reinforcement fine-tuning through difficulty-targeted online data selection and rollout replay. *Advances in Neural Information Processing Systems*, 38:173449–173476.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. TRL: Transformers Reinforcement Learning.

Xu Wan, Yansheng Wang, Wenqi Huang, and Mingyang Sun. 2026. Buffer matters: Unleashing the power of off-policy reinforcement learning in large language model reasoning. *arXiv preprint arXiv:2602.20722*.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. In *Advances in Neural Information Processing Systems*, volume 38, pages 115452–115486. Curran Associates, Inc.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Yifan Wei, Xiaoyan Yu, Yixuan Weng, Tengfei Pan, Angsheng Li, and Li Du. 2025. Autotir: Autonomous tools integrated reasoning via reinforcement learning. *arXiv preprint arXiv:2507.21836*.

Bo Wu, Sid Wang, Yunhao Tang, Jia Ding, Eryk Helenowski, Liang Tan, Tengyu Xu, Tushar Gowda, Zhengxing Chen, Chen Zhu, and 1 others. 2025. Llamarl: A distributed asynchronous reinforcement learning framework for efficient large-scale llm training. *arXiv preprint arXiv:2505.24034*.

Zhiheng Xi, Xin Guo, Yang Nan, Enyu Zhou, Junrui Shen, Wenxiang Chen, Jiaqi Liu, Jixuan Huang, Zhihao Zhang, Honglin Guo, and 1 others. 2025. Bapo: Stabilizing off-policy reinforcement learning for llms via balanced policy optimization with adaptive clipping. *arXiv preprint arXiv:2510.18927*.

Jie Xiao, Meng Chen, Qingnan Ren, Song Jingwei, Jiaqi Huang, Yangshen Deng, Chris Tong, Wanyi Chen, Suli Wang, Ziqian Bi, and 1 others. 2026. Echo-2: A large scale distributed rollout framework for cost-efficient reinforcement learning. *arXiv preprint arXiv:2602.02192*.

Xin Xu, Clive Bai, Kai Yang, Tianhao Chen, Yangkun Chen, Weijie Liu, Hao Chen, Yang Wang, Saiyong Yang, and Can Yang. 2026. Composition-rl: Compose your verifiable prompts for reinforcement learning of large language models. *arXiv preprint arXiv:2602.12036*.

Yixuan Even Xu, Yash Savani, Fei Fang, and J Zico Kolter. 2025. Not all rollouts are useful: Down-sampling rollouts in llm reinforcement learning. *arXiv preprint arXiv:2504.13818*.

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

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2026. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, and 1 others. 2026. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*.

Zhiyuan Zeng, Hamish Ivison, Yiping Wang, Lifan Yuan, Shuyue Stella Li, Zhuorui Ye, Siting Li, Jacqueline He, Runlong Zhou, Tong Chen, and 1 others. 2025. Rlve: Scaling up reinforcement learning for language models with adaptive verifiable environments. *arXiv preprint arXiv:2511.07317*.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhongzhi Li, Xiangyuan Xue, Yijiang Li, Yifan Zhou, Yang Chen, Chen Zhang, Yutao Fan, Zihu Wang, Songtao Huang, Francisco Piedrahita-Velez, Yue Liao, Hongru Wang, and 6 others. 2026a. The landscape of agentic reinforcement learning for llms: A survey. *Preprint*, arXiv:2509.02547.

Han Zhang, Ruibin Zheng, Zexuan Yi, Zhuo Zhang, Hanyang Peng, Hui Wang, Zike Yuan, Cai Ke, Shiwei Chen, Jiacheng Yang, and 1 others. 2025a. Gepo: Group expectation policy optimization for stable heterogeneous reinforcement learning. *arXiv preprint arXiv:2508.17850*.

Jiarui Zhang, Yuchen Yang, Ran Yan, Zhiyu Mei, Liyuan Zhang, Daifeng Li, Wei Fu, Jiaxuan Gao, Shusheng Xu, Yi Wu, and 1 others. 2026b. Areal-dta: Dynamic tree attention for efficient reinforcement learning of large language models. *arXiv preprint arXiv:2602.00482*.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, Yu Fu, Xingtai Lv, Yuchen Zhang, Sihang Zeng, Shang Qu, Haozhan Li, Shijie Wang, Yuru Wang, Xinwei Long, and 20 others. 2025b. A survey of reinforcement learning for large reasoning models. *Preprint*, arXiv:2509.08827.

Ruiqi Zhang, Daman Arora, Song Mei, and Andrea Zanette. 2025c. Speed-rl: Faster training of reasoning models via online curriculum learning. *arXiv preprint arXiv:2506.09016*.

Zhi Zhang, Zhen Han, Costas Mavromatis, Qi Zhu, Yunyi Zhang, Sheng Guan, Dingmin Wang, Xiong Zhou, Shuai Wang, Soji Adeshina, and 1 others. 2026c. Train less, learn more: Adaptive efficient rollout optimization for group-based reinforcement learning. *arXiv preprint arXiv:2602.14338*.

Yang Zhao, Kai Xiong, Xiao Ding, Li Du, Yangou Ouyang, Zhouhao Sun, Jiannan Guan, Wenbin Zhang, Bin Liu, Dong Hu, and 1 others. 2026. Ufo-rl: Uncertainty-focused optimization for efficient reinforcement learning data selection. *Advances in Neural Information Processing Systems*, 38:577–596.

Haizhong Zheng, Yizhuo Di, Jiahui Wang, Shuowei Jin, Xueshen Liu, Yongji Wu, Z Morley Mao, Ion Stoica, Jiawei Zhao, and Beidi Chen. 2026a. Astraflow: Dataflow-oriented reinforcement learning for agentic llms. *arXiv preprint arXiv:2605.15565*.

Haizhong Zheng, Jiawei Zhao, and Beidi Chen. 2025. Prosperity before collapse: How far can off-policy rl reach with stale data on llms? *arXiv preprint arXiv:2510.01161*.

Haizhong Zheng, Yang Zhou, Brian Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. 2026b. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts. *Advances in Neural Information Processing Systems*, 38:124321–124346.

Yuzhen Zhou, Jiajun Li, Yusheng Su, Gowtham Ramesh, Zilin Zhu, Xiang Long, Chenyang Zhao, Jin Pan, Xiaodong Yu, Ze Wang, Kangrui Du, Jialian Wu, Ximeng Sun, Jiang Liu, Qiaolin Yu, Hao Chen, Zicheng Liu, and Emad Barsoum. 2025. April: Active partial rollouts in reinforcement learning to tame long-tail generation. *Preprint*, arXiv:2509.18521.

Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. 2025. slime: An llm post-training framework for rl scaling. `https://github.com/THUDM/slime`. GitHub repository. Corresponding author: Xin Lv.

# A Experimental Setup and Implementation Details

To ensure reproducibility and to rigorously isolate the algorithmic impact of off-policy staleness, this section details the dataset construction, training protocols, hyperparameter configurations, and evaluation metrics used in our empirical case study.

## A.1 Dataset and Reward Formulation

The training data for our diagnostic experiments is sourced from the open-source **DeepScaler** dataset. We randomly sample 800 mathematical reasoning prompts for the main experiment. For each prompt, we sample 8 distinct responses ($N = 8$). With a prompt batch size of 8, each training batch comprises 64 responses.

To accommodate complex Chain-of-Thought (CoT) reasoning, the maximum generation length is set to 16,384 tokens. For the reward formulation, we strictly employ a *binary reward mechanism*: a reward of 1 is granted exclusively for an exact match with the ground-truth answer, and 0 otherwise. We deliberately exclude Process Reward Models (PRMs) or any heuristic step-level rewards to avoid confounding the gradient signals.

## A.2 Model and System Infrastructure

All experiments utilize the official **Qwen3-1.7B** (Yang et al., 2025) as the base model. At the system level, distributed training is implemented using the **veRL** (Sheng et al., 2024) reinforcement learning framework, coupled with **vLLM** (Kwon et al., 2023) as the backend rollout engine to maximize sampling throughput. All experiments are executed on a single compute node equipped with 8 $\times$ NVIDIA A100 (80GB) GPUs.

### A.3 Frozen-Trajectory Replay Protocol

To completely eliminate sampling variance and purely diagnose the effects of policy staleness, we design a controlled *frozen-trajectory replay protocol* consisting of two distinct phases:

- **Phase 1: On-Policy Buffer Construction (Epoch 1).** The model is trained using standard on-policy GRPO. During this epoch, all generated rollout trajectories and intermediate model checkpoints are cached.
- **Phase 2: Trajectory Replay (Epochs 2–4).** The online trajectory generation module is completely disabled. All compared methods resume training using strictly the same historical trajectories cached during Phase 1.

**Crucial Control Variable:** During the replay phase, data shuffling is strictly prohibited. This ensures that all evaluated methods encounter the exact same data distribution in an identical sequence. Consequently, any observed performance divergence is solely attributable to how each optimization objective handles stale trajectories.

### A.4 Baselines and Hyperparameters

We benchmark three distinct algorithmic settings in our replay experiment:

1. **On-policy GRPO:** A standard synchronous baseline generating fresh rollouts continually.
2. **Default Clip:** Naive trajectory replay employing standard PPO-style trust-region clipping, with the clip range set to $\epsilon = 0.2$.
3. **No Clip:** Retains importance sampling weights but removes all clipping constraints, serving as a diagnostic baseline to observe optimization collapse under unconstrained off-policy variance.

Across all experiments, the learning rate is fixed at $1 \times 10^{-6}$ and the mini-batch size is set to 8. The total training spans 4 epochs (1 on-policy epoch followed by 3 replay epochs). *Note:* In the main diagnostic experiments, the explicit KL divergence penalty is disabled to prevent it from masking the inherent pathologies of the off-policy correction mechanisms. For KL regularization ablation studies, the reference policy is anchored to the initially loaded model.

### A.5 Evaluation Protocol

Model mathematical reasoning capabilities are evaluated across five authoritative benchmarks: AIME24, AIME25, AMC23, MATH500 (Hendrycks et al., 2021), and OlympiadBench (He et al., 2024).

During training, evaluation is triggered every 20 steps. For decoding, we apply a sampling temperature of $0.6$ and a top-$k$ of $45$. To account for the extreme difficulty and high variance inherent in the AIME competitions, we report the mean@10 metric for AIME24 and AIME25. For AMC23, MATH500, and OlympiadBench, we report the standard mean@1 metric.

| **Hyperparameter** | **Value** |
|---|---|
| Base Model | Qwen3-1.7B |
| Hardware | $8 \times$ A100-80G GPUs |
| Software Stack | veRL + vLLM |
| Learning Rate | $1 \times 10^{-6}$ |
| Total Epochs | 4 (1 On-policy + 3 Replay) |
| Prompt Batch Size | 8 |
| Responses per Prompt ($N$) | 8 |
| Mini-batch Size | 8 |
| Max Generation Length | 16,384 tokens |
| Reward Formulation | Strict Binary (Correct/Incorrect) |
| KL Penalty | Disabled (in main diagnostic exp.) |
| Clip Range ($\epsilon$) | 0.2 (for Default Clip baseline) |
| **Evaluation Settings** | |
| Temperature | 0.6 |
| Top-$k$ | 45 |
| Evaluation Frequency | Every 20 training steps |

Table 1: Summary of hyperparameters and experimental settings.