NeurIPS 2026 · Poster

HIT SCIR · Peking University · Huawei

Unveiling Entropy–Performance Decoupling in Agentic RL for Tool-Integrated Reasoning

Yirong Zeng♪‡ · Shen You♪ · Yufei Liu♥ · Hao Cong♣ · Yutai Hou♩ · Xiao Ding♪* · Yuxian Wang♩ · Wu Ning♩ · Wang Xu♩ · Bibo Cai♪* · Dandan Tu♩ · Wenyu Zang♪♦ · Ting Liu♪

♪ Harbin Institute of Technology (SCIR Lab) · ♥ Peking University · ♣ Tsinghua University · ♩ Huawei Technologies Co., Ltd. · ♦ ‡ Interned at Huawei

Code: github.com/zeng-yirong/EarlyTIR

TL;DR — In text-only RL, entropy falls as accuracy converges. In agentic RL for TIR it does not: accuracy plateaus while entropy keeps climbing — and the extra uncertainty **buys no accuracy**. The driver is **invalid tool calls** trapping agents in retry loops. **EarlyTIR** truncates those rollouts → **+13.6% accuracy** (vs. base) and **−33.7% tool calls**.

1 Background & Motivation

- **Tool-Integrated Reasoning (TIR)** lets LLMs solve multi-step problems by offloading computation to external tools (code interpreters, APIs).
- **Agentic RL** (RLVR, e.g. DAPO) now drives TIR — but the **policy entropy dynamics** behind it remain a black box.
- In **text-only RL**, entropy decays exponentially as performance converges. TIR breaks this: entropy is shaped by **discrete external information gains** from environment feedback.

Key question: does the text-only entropy–performance relationship still hold once tool feedback enters the loop?

2 Finding 1 — Entropy–Performance Decoupling

Pre-study: 5 open-source models (Qwen2.5 / Qwen3, 7B–32B), trained with **DAPO** on DAPO-Math-17k + DeepCoder, evaluated on 6 benchmarks.

- **Early (< 50 steps):** entropy and accuracy rise together — healthy exploration of the hybrid action space.
- **Late:** accuracy plateaus while entropy **keeps diverging** — policy uncertainty becomes **ineffective**.

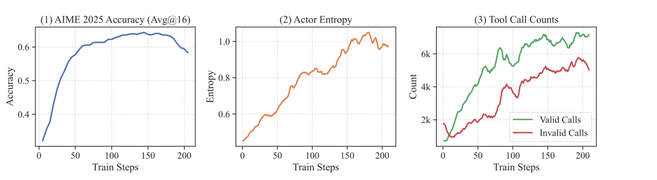

Fig. 1 (Qwen3-32B): accuracy plateaus while entropy keeps climbing and invalid tool calls surge — the decoupling signature.

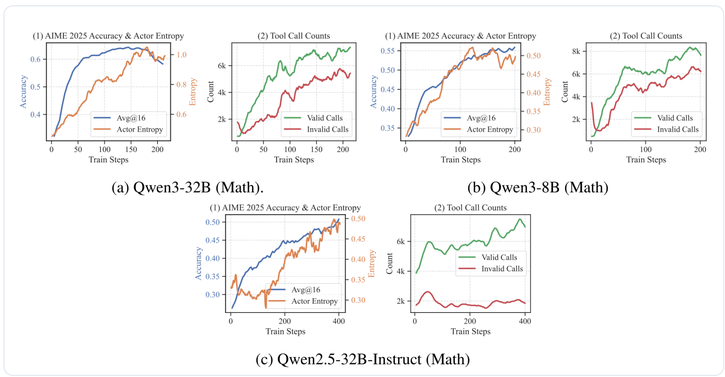

Fig. 2: the same pattern at 8B and 32B, on Qwen3 and Qwen2.5. Qwen2.5-32B-Instruct is the exception — invalid calls stay low and accuracy keeps improving.

3 Finding 2 — Invalid Tool Calls Drive It

Invalid calls = no information gain: execution errors, timeouts, empty returns, redundant commands. They are not noise — they are the mechanism.

- Entropy grows much faster at high invalid-call ratios — Spearman = 0.93 ($p < 0.05$) between invalidity bins and terminal entropy.
- The gap widens with training: invalid calls are a **primary driver**, not a side effect.
- Accuracy peaks at **3–6 tool turns** and collapses past 10, where **> 75%** of calls are invalid.

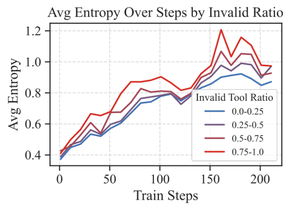

Fig. 3: entropy over steps, grouped by invalid-call ratio.

Error-correction trap: once a sub-problem survives a few retries, the agent loops on invalid tools, parameters and APIs — high uncertainty, low utility.

4 EarlyTIR: Early Intervention for TIR

A **strategic truncation mechanism** that prunes non-informative exploration during the **RL rollout** — before hallucination loops corrupt the context window.

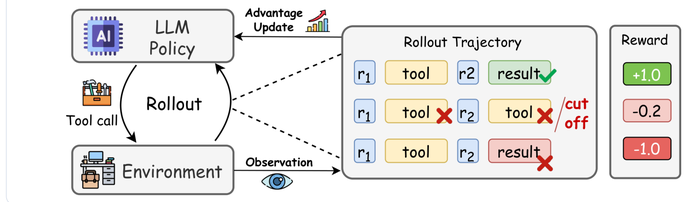

Fig. 4: a rollout that exceeds the invalid-call threshold τ is halted and penalized.

1. **Monitor.** A rule-based detector flags 4 objectively verifiable failure modes from the sandbox: **execution errors**, **timeouts**, **empty returns**, **redundant commands**. (50-trajectory audit: **100% precision**.)
2. **Truncate.** Once cumulative invalid calls exceed tolerance τ, generation halts immediately — the loop breaks and noisy feedback never enters the context window.
3. **Penalize & learn.** The truncated path receives $R_t = -0.2$ and is kept as a **negative sample** in the DAPO loss; τ > 0 preserves **error-recovery** learning from occasional early mistakes.

Why tolerate errors? Very short trajectories (< 2 turns) under-solve; peak performance needs a **tolerance window**, not blanket prohibition.

5 Experimental Setup

Algorithm: DAPO (RLVR) on GRPO · **Framework:** veRL · **Tool:** Python interpreter, local sandbox
Models: Qwen2.5 7B/32B (SFT → RL) · Qwen3 8B/14B/32B (RL directly)
Baselines: ReTool, SimpleTIR, ZeroTIR, DeepCoder, KodCode, MicroCoder
Eval: AIME 24/25, HMMT 25, AMO-Bench (math) · HumanEval, LiveCodeBench v6 (code)
EarlyTIR: τ = 2 · $R_t = -0.2$ · batch 512 · group 16 · PPO mini-batch 128 · 16K tokens · 8 tool turns

6 Results — Math & Coding

+13.6% macro-average accuracy over the Qwen3-32B base (54.22 vs 40.59)

−33.7% tool calls per sample at evaluation (avg. over 6 benchmarks)

Model	AIME24	AIME25	HMMT25	AMO-B.	Avg
Qwen3-8B base	63.33	50.33	43.33	4.60	40.40
♥ ReTool-8B	64.17	57.08	44.76	5.75	42.94
♥ EarlyTIR-8B	**67.08**	**60.42**	**46.67**	**7.75**	**45.48**
Qwen3-32B base	70.37	54.81	29.17	8.00	40.59
♥ ReTool-32B	70.00	61.11	38.89	6.67	44.17
♥ EarlyTIR-32B	**82.50**	**69.58**	**52.00**	**12.80**	**54.22**
Qwen2.5-7B-inst	7.33	5.33	3.67	1.20	4.38
♥ SimpleTIR-7B	50.50	30.90	29.70	–	–
♥ ZeroTIR-7B	39.60	25.00	22.50	–	–
♥ ReTool-7B	40.00	23.33	–	–	–
♥ EarlyTIR-7B	**42.00**	**31.25**	**15.83**	**0.75**	**22.46**

EarlyTIR-32B beats the strongest TIR baseline (ReTool) by **+10.05** absolute macro-average and leads every specialized method; on Qwen2.5-7B it lifts the average from 4.38 to **22.46**. Gains hold from 7B to 32B.

Coding Benchmarks (* = trained on consistent data subsets)

Model	HumanEval	LiveCodeBench
Qwen3-8B base	81.27	34.80
♥ DeepCoder*	90.60	39.70
♥ KodCode	89.31	33.46
♥ MicroCoder	94.38	42.73
♥ EarlyTIR-8B*	**93.40**	**42.80**
Qwen3-32B base	89.94	42.75
♥ DeepCoder*	92.37	43.11
♥ EarlyTIR-32B*	**94.02**	**47.27**

7 Training Dynamics & Efficiency

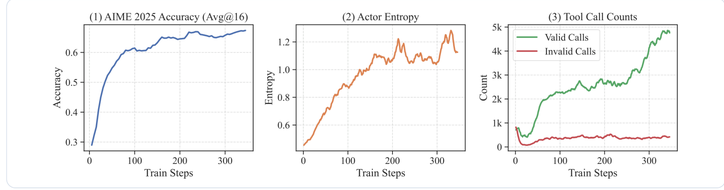

Fig. 5 (Qwen3-32B + EarlyTIR): accuracy rises without late-stage decay; entropy converges *with* accuracy — no decoupling; invalid calls stay near zero.

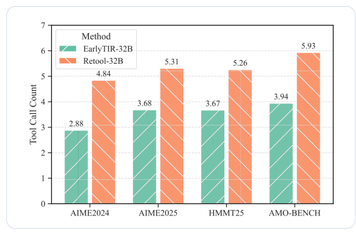

Fig. 6: average tool calls per sample at evaluation.

- Tool use cut by **30.2–40.5%** per benchmark (avg. **−33.7%**).
- AMO-Bench: 5.93 → **3.94** calls per sample.
- Lower latency and environment overhead per solved problem.

Not prompt steering: told to "use tools only if absolutely necessary," ReTool loses 4.73 points on AIME24 but EarlyTIR only 1.12 — the policy has **learned** when to call a tool.

8 Ablation Studies (Qwen3-32B)

τ Tolerance: non-monotonic. τ = 2 is best overall (82.50 / 69.58); τ = 4 wins on the harder AIME25. Recipe: **2** for simple tasks, **4** for hard ones.

Rt Penalty: mild $R_t = -0.2$ works best; too severe (−1.0) suppresses tool use and falls back to base-model level.

✓ Truncation is the ingredient: penalizing without halting (No-Truncat) gains only +1.24 on AIME24; EarlyTIR gains **+12.50 / +8.47** — the context must be freed from retry loops.

Setting (Qwen3-32B)	AIME24	AIME25
ReTool (baseline)	70.00	61.11
Tolerance threshold τ		
w/ τ = 2 (default)	**82.50**	**69.58**
w/ τ = 3	72.92	60.83
w/ τ = 4	78.89	70.00
w/ τ = 5	80.00	64.44
No-Truncat (penalty only)	71.24	60.87
Penalty reward R_t		
w/ $R_t = -0.2$	82.50	69.58
w/ $R_t = -0.5$	71.11	61.11
w/ $R_t = -1.0$	70.00	65.56

+ Limitations & Future Directions

- **Adaptive thresholds:** the optimal τ tracks task difficulty; adaptive selection is future work.
- **Scope:** the rule-based detector covers limited objectively verifiable failures, not semantic redundancy.

9 Conclusions & Takeaways

- **Entropy–performance decoupling** is pervasive in agentic RL for TIR: entropy diverges while accuracy stagnates.
- It is driven by **accumulating invalid tool-use trajectories** that trap agents in non-informative loops.
- **EarlyTIR** — simple, rule-based, model-agnostic — breaks the loops while preserving error-recovery learning.
- Result: a **higher accuracy ceiling** (+13.6% over base) and **cheaper inference** (−33.7% tool calls).

40th Conference on Neural Information Processing Systems (NeurIPS 2026)