

LLMs Can Learn to Reason Via Off-Policy RL (OAPL)

2026-03-01

Paper Info

- **Authors:** Daniel Ritter, Owen Oertell, Bradley Guo, Jonathan Chang, Kianté Brantley, Wen Sun (Cornell, Databricks, Harvard)
- **Link:** [arXiv](#)
- **Read date:** 2026-03-01

Summary

Modern RL post-training (e.g. GRPO) assumes on-policy data, but in practice the inference engine (vLLM) and trainer (HuggingFace) often disagree on log-probs or run asynchronously, making data effectively off-policy. Prior work either adds importance sampling (high variance) or modifies the inference engine (slow, doesn't fix async lag). OAPL embraces off-policyness: it formulates a KL-regularized objective where the KL is to the lagged inference policy π_{vllm} , derives a closed-form optimal policy, and trains with a simple squared regression loss that fits $\beta \ln(\pi/\pi_{\text{vllm}})$ to the optimal advantage. No importance weights, no clipping. Sync trainer and inference infrequently (e.g. every 50–400 steps); train off-policy in between. Outperforms GRPO+IS on math (AIME, HMMT, BRUMO), matches DeepCoder on LiveCodeBench with $\sim 3\times$ fewer samples, and scales better on Pass@k.

Key Ideas

1. **On-policy is not necessary** — off-policy methods (DDPG, SAC) often beat PPO/REINFORCE in classic RL; same applies to LLM post-training.

2. **KL to π_{vllm} instead of π_{old}** — GRPO clips π/π_{old} but the first gradient can already push π far away; OAPL explicitly regularizes against the sampling policy.
3. **Closed-form optimum** — $\pi^*(y|x) \propto \pi_{\text{vllm}}(y|x) \exp(r/\beta)$, so $\beta \ln(\pi^*/\pi_{\text{vllm}}) = A^* = r - V^*$.
4. **$\hat{V}^*$ from group rollouts** — estimate $V^*(x) = \beta \ln \frac{1}{G} \sum_i \exp(r(x, y_i)/\beta)$ using rollouts from π_{vllm} ; no separate critic.
5. **Squared regression loss** — minimize $\sum_{x,i} (\beta \ln \frac{\pi(y_i|x)}{\pi_{\text{vllm}}(y_i|x)} - (r(x, y_i) - \hat{V}^*(x)))^2$. Same minimizer regardless of sampling distribution.

Method

- **Objective (Eq. 1):** $\max_{\pi} \mathbb{E}_{y \sim \pi} r(x, y) - \beta \text{KL}(\pi \| \pi_{\text{vllm}})$ — maximize reward, stay close to inference policy.
- **Optimal advantage:** $A^* = r - V^*$, with $V^* = \beta \ln \mathbb{E}_{y \sim \pi_{\text{vllm}}} \exp(r/\beta)$.
- **Algorithm:** Sync π and π_{vllm} ; generate rollouts from π_{vllm} ; add to buffer; train π on Eq. 3; every L steps, sync and clear buffer.
- **Two betas:** β_1 for $\hat{V}^*$ (Eq. 2), β_2 for the loss (Eq. 3). $\beta_1 = 1$, $\beta_2 = 1e - 3$ worked well.
- **Policy lag L :** Math: 50 or 100; code: ~ 400 gradient steps per sync.

Results

- **Math (Qwen3-4B-Thinking):** OAPL outperforms GRPO+IS on AIME-25, HMMT-25 (Feb/Nov), BRUMO-25 across Pass@1/5/10.
- **Code (DeepSeek-R1-Distill-Qwen-14B):** Matches DeepCoder on LiveCodeBench Pass@k with $\sim 200\text{K}$ vs $\sim 650\text{K}$ training samples ($\sim 3\times$ fewer).
- **Entropy:** OAPL preserves entropy; GRPO collapses it. OAPL scales better on Pass@k for $k = 1..256$.
- **Robustness:** Stable with $L = 100$ (math) and $L \approx 400$ (code) — $100\times$ more off-policy than prior IS-based work.

Strengths

- Principled off-policy objective with no importance sampling; avoids variance and heuristics (clip ratios, delete tokens).
- Simple squared regression; no critic, no extra value network.
- Fully asynchronous between syncs; reuse data, fewer generations.
- Empirically: better stability, entropy, and Pass@k scaling than GRPO.
- Directly addresses trainer–inference mismatch (FP differences, async pipelines).

Weaknesses / Limitations

- Two-stage code setup filters prompts with no correct solutions; may not reflect real online training.
- β_1, β_2 swept for math but not for code; sensitivity under-studied.
- Compared mainly to GRPO+IS; fewer comparisons to REBEL, REFUEL, AGRO, or other squared-loss methods.
- No ablations on L ; unclear sweet spot for different scales/tasks.

Relevance to My Work

- **MedEnv / GRPO:** Same verifiable-reward setting (math correctness, code execution). If our trainer and inference diverge (async, different backends), OAPL could be a drop-in alternative to GRPO+IS.
- **Sample efficiency:** $3\times$ fewer generations for similar performance — matters for expensive clinical trajectory rollouts.
- **Entropy and Pass@k:** If we care about Pass@k for clinical plan generation (sample k plans, take best), OAPL’s better scaling is directly relevant.
- **No importance weights:** Simpler, more stable than adding IS to GRPO when we have policy lag.

Key Takeaways

1. Off-policy RL is viable and often preferable for LLM post-training when trainer and inference disagree or lag.
2. KL to π_{vllm} + squared regression to A^* yields a simple, stable objective.
3. Infrequent sync ($L = 50\text{--}400$) + explicit KL regularization preserves entropy and improves Pass@k scaling vs GRPO.
4. OAPL matches or beats GRPO with $\sim 3\times$ fewer training samples on code.