PR #2290 完整报告

radixark/miles

fix(ci): persist every step metric for historical gate

合并时间：2026-08-09 14:03

原文链接：http://prhub.com.cn/radixark/miles/pull/2290

执行摘要

- 一句话：标准 RL 指标默认全 step 落库，历史门禁不再丢步骤
- 推荐动作：值得快速阅读（全 PR 仅 53 行新增、9 行删除），尤其适合 CI/metric-history 门禁的维护者。重点看三处：GATE_DEFAULTS 默认值表、坐标键（steps_key）推导与基线冷启动的联动、TITO 保持 "last" 的设计理由。不涉及训练核心代码，无需精读生产路径。

功能与动机

PR body 给出了明确的症状与复现：register_ci_gate(metric_key="train/train_rollout_logprob_abs_diff") 只持久化了序列的最后一个点（step=-1），导致 step 级历史不可用；一个 4 点 nightly 记录经过单行标准 RL 门禁只产生 1 行 metric_values。根因链为：GATE_DEFAULTS 对标准 RL 指标赋 steps="last" → select_metric_values 把序列归约到最后一点 → write_run 以 step=-1 落库。而文档对 steps="all" 的语义定义是每个 step 独立与该 step 自身历史比较，因此旧默认值与 metric-history 门禁的设计意图相悖，属于功能性缺陷。

实现拆解

1. 根因定位（tests/ci/metric_history/register.py）：GATE_DEFAULTS 表中 5 个标准 RL 指标（train/grad_norm、train/ppo_kl、train/train_rollout_logprob_abs_diff、train/train_rollout_kl、rollout/raw_reward）的默认 steps 均为 "last"，是整个缺陷的源头。
2. 默认值修改（同一文件）：5 个指标的 steps 改为 "all"，让选择与落库保留每个 step 的独立坐标（steps_key 从 '"last"' 变为 '"all"'）；TITO 两个条目（rollout/tito_session_mismatch_rate/v1|v2/assistant_text）保持 "last" 不变，显式声明的 steps/constraint 仍在解析时优先于表内默认值。
3. 选择层测试（tests/ci/test/test_metric_selection.py）：test_one_liner_fills_from_gate_defaults 与 test_partial_default_written_literal_wins 的断言从 steps == "last" / steps_key == '"last"' 更新为 "all" 版本；新增参数化测试 test_standard_rl_gate_defaults_capture_all_steps 逐个锁定 5 个指标的默认值，防止改表时漏改。
4. 集成层测试（tests/ci/test/test_gate_integration.py）：新增 test_nightly_one_liner_writes_every_step，用 4 个 step 的 rollout/raw_reward 记录走 nightly 全流程，断言 metric_values 表产生 step 0–3 共 4 行且 steps_key 均为 '"all"'，直接验证持久化行为。

5. 文档同步（docs/ci/03-metric-history-gate.md）：更新 "Defaults for standard metrics" 说明，明确标准 RL 指标默认 steps="all"；新增 "Default calibration and resets" 小节，说明改表字面量会 re-key 默认坐标并导致基线冷启动。 全部配套测试 tests/ci/test 运行结果：325 passed、1 skipped；无生产代码、配置或部署配套改动。

关键文件：

- tests/ci/metric_history/register.py（模块 指标注册；类别 test；类型 configuration；符号 GATE_DEFAULTS）：本次修复的实际落点：GATE_DEFAULTS 表中 5 个标准 RL 指标的默认 steps 从 "last" 改为 "all"，TITO 条目保持 "last"，直接决定 nightly 落库的分 step 行为。
- tests/ci/test/test_gate_integration.py（模块 门禁集成；类别 test；类型 test-coverage；符号 test_nightly_one_liner_writes_every_step）：新增集成测试 test_nightly_one_liner_writes_every_step，从 nightly 全流程验证每个 step 独立落库，是修复行为的最直接证据。
- tests/ci/test/test_metric_selection.py（模块 指标选择；类别 test；类型 test-coverage；符号 test_standard_rl_gate_defaults_capture_all_steps, test_one_liner_fills_from_gate_defaults, test_partial_default_written_literal_wins）：更新两处默认值断言并新增参数化测试，锁定 5 个标准 RL 指标的默认 steps 为 "all"，防止后续改表时漏改。
- docs/ci/03-metric-history-gate.md（模块 CI 文档；类别 docs；类型 documentation）：同步更新门禁文档，明确标准 RL 指标默认 steps="all"，并新增默认值校准与基线冷启动说明。

关键符号：test_nightly_one_liner_writes_every_step, test_standard_rl_gate_defaults_capture_all_steps, test_one_liner_fills_from_gate_defaults, test_partial_default_written_literal_wins

关键源码片段

tests/ci/metric_history/register.py

本次修复的实际落点：GATE_DEFAULTS 表中 5 个标准 RL 指标的默认 steps 从 "last" 改为 "all"，TITO 条目保持 "last"，直接决定 nightly 落库的分 step 行为。

```
# tests/ci/metric_history/register.py
# GATE_DEFAULTS 表用于补全单行声明 register_ci_gate(metric_key=...) 的 steps
# 与 constraint 字面量。这些字面量参与与手写字面量相同的坐标推导：修改表内
# 值会重新生成依赖它的声明的坐标键（steps_key / constraint_key），并导致
# 对应基线冷启动（见 docs 的 Identity 一节）。
GATE_DEFAULTS: dict[str, dict] = {
    # 标准 RL 指标默认 steps="all"，每个 step 都按独立坐标落库，历史门禁
    # 可以按训练步与自身历史一一比较，而不是只看最后一个点。
    "train/grad_norm": {
        "steps": "all",
        "constraint": {"rel_up": 0.5, "abs_floor_up": 0.1, "rel_down": 0.8, "abs_floor_down": 0.1},
    },
    "train/ppo_kl": {
```

```
        "steps": "all",
        "constraint": {"rel_up": 0.5, "abs_floor_up": 0.02, "rel_down": 0.8, "abs_floor_down": 0.02},
    },
    "train/train_rollout_logprob_abs_diff": {
        "steps": "all",
        "constraint": {"rel_up": 0.5, "abs_floor_up": 0.02, "rel_down": 0.8, "abs_floor_down": 0.02},
    },
    "train/train_rollout_kl": {
        "steps": "all",
        "constraint": {"rel_up": 0.5, "abs_floor_up": 0.02, "rel_down": 0.8, "abs_floor_down": 0.02},
    },
    "rollout/raw_reward": {
        "steps": "all",
        "constraint": {"rel_up": 0.5, "abs_floor_up": 0.05, "rel_down": 0.2, "abs_floor_down": 0.05},
    },
    # TITO（session 会话错配率）保持 steps="last"，它是整个会话周期的汇总
    # 指标，按最后一点比较更贴合语义，不受本次变更影响。
    "rollout/tito_session_mismatch_rate/v1/assistant_text": {
        "steps": "last",
        "constraint": {"rel_up": 1.0, "abs_floor_up": 0.1, "rel_down": 1.0},
    },
    "rollout/tito_session_mismatch_rate/v2/assistant_text": {
        "steps": "last",
        "constraint": {"rel_up": 1.0, "abs_floor_up": 0.1, "rel_down": 1.0},
    },
}
```

tests/ci/test/test_gate_integration.py

新增集成测试 test_nightly_one_liner_writes_every_step，从 nightly 全流程验证每个 step 独立落库，是修复行为的最直接证据。

```
# tests/ci/test/test_gate_integration.py
# 单行声明不写 steps / constraint，解析时从 GATE_DEFAULTS 取默认值 "all"；
# nightly 落库时每个 step 都必须独立成行，而不是只保留最后一点。
def test_nightly_one_liner_writes_every_step(self, tmp_path, store):
    test_file = _write_test_file(
        tmp_path,
        'register_ci_gate(metric_key="rollout/raw_reward")',
    )
    registry = _registry(test_file)
    # 构造 4 个 step 的指标序列，step 0–3 对应 0.20 / 0.40 / 0.60 / 0.80
    record = _write_record(
        tmp_path,
        {"rollout/raw_reward": [[0, 0.20], [1, 0.40], [2, 0.60], [3, 0.80]]},
        name="m.jsonl",
    )

    run_gate_hook(
        test_file,
```

```
        record,
        store=store,
        registry=registry,
        nightly=True,
        provenance=PROVENANCE,
    )

    # 断言 metric_values 表按 step 升序出现 4 行，坐标 steps_key 均为 '"all"'
    rows = store._conn.execute("SELECT steps_key, step, value FROM metric_values ORDER BY
    step").fetchall()
    assert rows == [
        ('"all"', 0, 0.20),
        ('"all"', 1, 0.40),
        ('"all"', 2, 0.60),
        ('"all"', 3, 0.80),
    ]
```

tests/ci/test/test_metric_selection.py

更新两处默认值断言并新增参数化测试，锁定 5 个标准 RL 指标的默认 steps 为 "all"，防止后续改表时漏改。

```
# tests/ci/test/test_metric_selection.py
# 参数化锁定 5 个标准 RL 指标在 GATE_DEFAULTS 中的默认 steps 均为 "all"，
# 防止后续改表时漏改或误改导致单行声明行为回退。
@pytest.mark.parametrize(
    "metric_key",
    [
        "train/grad_norm",
        "train/ppo_kl",
        "train/train_rollout_logprob_abs_diff",
        "train/train_rollout_kl",
        "rollout/raw_reward",
    ],
)
def test_standard_rl_gate_defaults_capture_all_steps(metric_key):
    assert GATE_DEFAULTS[metric_key]["steps"] == "all"
```

评论区精华

本 PR 没有任何 review 评论（comments 与 review_comments 均为 0），未发生实质性讨论。作者在 PR body 中自列了 3 个 Review Focus，代表本变更最值得审视的点：一是 5 个 GATE_DEFAULTS 条目改为 steps="all" 是否合理；二是 TITO 为什么保留 steps="last"；三是集成测试的持久化断言是否覆盖到位。其中 TITO 保持 "last" 的理由在代码与文档中已有交代：TITO 是 session 级错配率的会话汇总指标，天然按最后一点或整体值比较，不属于按训练步对照的场景。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 基线坐标冷启动（确定性风险）：steps_key 是存储值的身份组成部分（与 step、constraint_key 共同构成坐标），把默认值从 "last" 改为 "all" 会重新生成坐标键，历史基线中 '"last"' 坐标的记录不再参与比较，5 个指标的历史门禁需要重新积累基线；register.py 注释与文档均明确提示了这一后果。
  2. 门禁判定范围扩大：steps="all" 下每个 step 独立与该 step 自身历史比较，判定粒度从 " 整体最后一点 " 变为 " 逐 step"，波动较大的 step（如训练早期）可能带来更多 untrusted 结果，属于预期但需观察的行为变化。
  3. 落库数据量增长：nightly 每个指标从 1 行变为每 step 1 行，对 step 数大的运行，metric_values 表行数成倍增长，当前体量下影响有限。
  4. 变更不触碰 miles/utils/tracking_utils/ci_history.py 等生产路径，无训练 /rollout 运行时风险。 - 影响：影响范围集中在 CI metric-history 门禁子系统（tests/ci 与 docs/ci），对使用 5 个标准 RL 指标单行声明的所有 nightly/test 用例生效；不涉及模型训练、rollout 等运行时行为。正向影响是历史门禁第一次具备逐训练步比较能力，能更早暴露特定 step 的指标漂移。操作影响是上述 5 个指标的既有 '"last"' 基线将冷启动，TITO 门禁行为不变。对团队而言，维护 CI 指标门禁的工程师需要理解：修改 GATE_DEFAULTS 表字面量会 re-key 坐标并冷启动基线。 - 风险标记：基线坐标冷启动，门禁判定范围扩大，落库行数增长

关联脉络

- PR #2218 ci: version TITO metrics by session server: 与本 PR 直接修改同一张 GATE_DEFAULTS 表（tests/ci/metric_history/register.py）与同一份选择层测试（tests/ci/test/test_metric_selection.py），引入 TITO v1/v2 门禁默认条目（steps="last"）；本 PR 新增 5 个标准 RL 指标默认 steps="all" 并明确保留 TITO 的 "last" 策略，是该门禁默认值体系的延续与修正。