

PR #2881 完整报告

radixark/miles

Dashboard: handle multiple TITO leaves per sample index

合并时间: 2026-09-03 04:35

原文链接: <http://prhub.com.cn/radixark/miles/pull/2881>

执行摘要

PR#2881 修复了 Dashboard 在处理 TITO v2 训练框架生成的共享 `Sample.index` 叶子样本时的关键缺陷。通过引入 `(sample_index, sample_occurrence)` 二元组作为新的数据键，从后端数据加载、API 到前端路由与视图进行了全链路适配，解决了因数据混淆导致的 API 500 崩溃问题，并确保每个叶子样本的可独立查看性。修复同时兼顾了向后兼容性和历史数据支持。

功能与动机

如 PR body 所述，问题根源在于：“TITO v2 can emit multiple leaf samples that intentionally share one `Sample.index`。The dashboard previously keyed all rollout and train data only by that index, conflating distinct leaves。”在一次真实运行中，这直接导致了 `/api/rollout/549/summary` 接口返回 HTTP 500 错误，因为两个属于同一 `sample_index=35147` 的 rank-0 行具有不同的 `response_length`。本次变更旨在从根本上解决 Dashboard 的数据模型假设，使其能正确支持多叶子样本场景。

实现拆解

- 核心数据模型与 Join 逻辑重构：在 `miles/dashboard/dump_reader.py` 中，引入 `SampleKey = tuple[int, int]` 类型。新增 `_sample_keys()` 函数，为每个样本计算在其所属 rollout 步中的 `occurrence` 序号。新增 `_matching_train_locations()` 函数，用于在旧式训练数据转储中（仅记录 `sample_indices`），通过比对 `response_length`、`total_length` 和 `tokens` 内容来精准匹配 TITO 叶子样本对应的物理行，避免混淆。
`JoinedRollout.train_rows` 字典的键从 `int` 改为 `SampleKey`。`SUMMARY_VERSION` 从 5 升至 6 使旧摘要缓存自动失效，`SUMMARY_COLUMNS` 新增 `sample_occurrence` 列。
- API 层与后端数据生产适配：在 `miles/dashboard/server.py` 中，`sample_messages` 和 `sample_tokens` 接口新增 `occurrence` 查询参数（默认 0 以保持向后兼容）。在 `miles/ray/rollout/debug_data.py` 的 `trajectory_rows()` 函数中，现在为每个样本在侧车 JSONL 文件中计算并持久化正确的 `sample_occurrence`。
- 前端路由与视图更新：在 `miles/dashboard/static/app.js` 中，路由解析现在从 URL 参数中提取 `occurrence`。在 `miles/dashboard/static/views_rollout.js` 和 `miles/dashboard/static/views_tokens.js` 中，引入新的 `sampleHash()` 函数，用于生成包含 `occurrence` 参数的哈希路由。UI 元素如散点图标签、组导航面板均更新为使用完整的 `(sample_index, sample_occurrence)` 键进行定位和跳转。
- 测试覆盖：测试辅助模块 `tests/fast/dashboard/dummy_dump.py` 新增 `duplicate_first_sample_index` 参数以模拟多叶子场景。新增多个关键测试用例覆盖新逻辑

与降级路径: test_duplicate_sample_indices_keep_tito_leaves_distinct (验证核心 join)、test_partial_train_coverage_degrades_to_null_rows (验证训练数据缺失时降级为 null 行)、test_trajectory_occurrences_follow_the_full_sample_order (验证轨迹侧车文件编号)、test_duplicate_tito_leaves_are_independently_viewable (端到端 API 测试)。

miles/dashboard/static/views_tokens.js

Token 查看视图, 实现了携带 occurrence 参数的路由跳转和 UI 导航, 是前端适配的关键。

```
function sampleHash(rolloutId, row, evaluation) {
  const params = new URLSearchParams();
  // 仅在 occurrence 非零时添加到查询参数, 保持 occurrence=0 的向后兼容
  if (row.sample_occurrence) params.set("occurrence", String(row.sample_occurrence));
  if (evaluation) params.set("eval", "1");
  const query = params.toString();
  return `#/rollout/${rolloutId}/sample/${row.sample_index}${query ? `? ${query}` : ""}`;
}

async function groupNavPanel(rolloutId, sampleIndex, sampleOccurrence, evaluation) {
  const { rows } = await api(`/api/rollout/${rolloutId}/summary`, { eval: evaluation });
  // 根据完整的 (sample_index, sample_occurrence) 查找当前选中行
  const selected = rows.find(
    (row) => row.sample_index === sampleIndex && row.sample_occurrence ===
    sampleOccurrence,
  );
  const groupIndex = selected?.group_index;
  // 兄弟节点按 (sample_index, sample_occurrence) 排序
  const siblings = groupIndex == null
    ? []
    : rows
      .filter((row) => row.group_index === groupIndex)
      .sort((a, b) => a.sample_index - b.sample_index || a.sample_occurrence - b.sample_occurrence);
  if (siblings.length < 2) return null; // no group, or a lone sample: nothing to navigate
  const position = siblings.indexOf(selected);
  const goto = (row) => (location.hash = sampleHash(rolloutId, row, evaluation));
  // ... 其余导航按钮逻辑使用 goto(siblings[position - 1]) 等
}
```

miles/dashboard/static/views_rollout.js

Rollout 概览视图, 集成了新的 sample_occurrence 列显示和跳转逻辑。

```
function sampleHash(rolloutId, row, evaluation) {
  const params = new URLSearchParams();
  if (row.sample_occurrence) params.set("occurrence", String(row.sample_occurrence));
  if (evaluation) params.set("eval", "1");
  const query = params.toString();
  return `#/rollout/${rolloutId}/sample/${row.sample_index}${query ? `? ${query}` : ""}`;
}

// ... 在 renderRollout 中
```

```
const openTokens = (row) => (location.hash = sampleHash(rolloutId, row, evaluation));
// 在散点图标签中，清晰显示叶子编号
label: `sample ${r.sample_index}${r.sample_occurrence ? `#${r.sample_occurrence + 1}` : ""}
\n` +
  `reward=${fmtNum(r[rewardKey])} len=${r.response_length}` +
  (r.mean_abs_lp_diff !== null ? `\\nllp diffl=${fmtNum(r.mean_abs_lp_diff)}` : ""),
```

评论区精华

自动化代码审查工具 [claude\[bot\]](#) 在此次 review 中扮演了关键角色，指出了两个重要正确性问题：

"🔗 [load_joined](#) now does `locations = index[key[0]]` for every sample, a bare dict lookup that raises `KeyError` when a sample has no train row at all (partial `train_coverage`)... Fix: use `index.get(key[0], [])` and only build/assign a `TrainRow` when `locations` is non-empty, matching the base's tolerance for missing coverage." 此问题在后续提交中修复。

"🔗 [trajectory_messages\(\)](#) numbers occurrences by position within the trajectory jsonl file, but that file only contains rows for samples with `metadata["messages"]`... while every other occurrence numbering (summary rows, `_sample_keys`, `tokens`) counts over the full unfiltered sample list... Fix: derive trajectory occurrence from the full sample order (e.g. persist occurrence in the jsonl row)." 作者修复了此问题，[Zhichenzzz](#) 也对此分析表示同意。

风险与影响

- 技术风险：主要风险为 向后兼容性和 引入的逻辑复杂度。新引入的 `occurrence` 参数默认值为 0，确保了现有 URL 和 API 调用无需修改即可工作。SUMMARY_VERSION 的升级自动使旧缓存失效。`_matching_train_locations` 和侧车文件的回退逻辑确保了能处理旧格式的训练数据和轨迹文件。变更逻辑清晰，测试覆盖充分（249 passed），回归风险相对可控。
- 影响范围：影响范围集中在 `miles/dashboard` 模块及其依赖的数据读取（`dump_reader.py`）与生产（`debug_data.py`）部分。修复后，所有使用 TITO v2 进行训练并查看 Dashboard 数据的用户将受益，工具的可靠性和数据准确性得到提升。变更不涉及训练核心逻辑，对训练性能和稳定性无直接影响。

关联脉络

- 与历史 PR 关联：本 PR 与 PR#2817 (dashboard: give a step with no samples the summary schema) 同属提升 Dashboard 稳定性和数据准确性的系列工作。PR#2817 修复了空 step 的 schema 问题，本 PR 则修复了多叶子样本的数据混淆问题，两者共同强化了 Dashboard 面对复杂训练数据的鲁棒性。
- 演进趋势：此次变更揭示了 Dashboard 需要更灵活的数据键来适应越来越复杂的训练策略（如 TITO v2）。引入 `(sample_index, occurrence)` 二元组为未来可能的其他多叶子 / 多路径场景提供了可扩展的数据模型基础。