PR #2710 完整报告

radixark/miles

Log compaction-aware rollout metrics

合并时间：2026-08-25 10:38

原文链接：http://prhub.com.cn/radixark/miles/pull/2710

执行摘要

- 一句话：新增 compaction 感知的 rollout 训练样本与奖励指标
- 推荐动作：建议精读该 PR 以理解如何在 rollout 指标中正确处理 session compaction。值得关注的设计决策：奖励按 rollout 等权平均，避免长 rollout 权重过大；分组 key 的设计兼顾 rollout_id、index 和位置回退；使用 metadata 中的 raw_reward 进行奖励计算的策略。

功能与动机

Session compaction 会将一个 rollout 变成多个训练样本，导致现有 raw-reward 指标按样本加权，使长 rollout 获得不成比例的权重；同时 W&B 未暴露每步实际训练样本数。PR 旨在新增 rollout/num_training_samples 与 rollout/episode_raw_reward 指标，使这些影响可见，且不改变现有指标历史的含义。

实现拆解

1. 在 miles/ray/rollout/metrics.py 中新增 _compute_training_sample_metrics 函数：该函数遍历所有样本，以 (类型, group_index, 具体值) 作为 key 对样本分组，并维护一个 rewards_by_rollout 字典。分组 key 优先使用 (rollout, group_index, rollout_id)，其次使用 (sample, group_index, index)，最后回退到 (position, group_index, 枚举位置)。
2. 对每组奖励先求平均，得到每个 rollout 的等权平均奖励，再对所有 rollout 的平均值取平均，作为 episode_raw_reward；样本总数直接取 len(samples)。
3. 在 _compute_metrics_from_samples 中将该函数返回的指标合并进 log_dict，从而通过 log_rollout_data 统一输出并为 key 加 rollout/ 前缀。
4. 测试方面，在 tests/fast/ray/rollout/test_metrics.py 增加 TestTrainingSampleMetrics 测试类，覆盖 5 个场景；并在既有 test_rollout_log_fans_out_versioned_tito_keys 中增加对新增指标键的断言。测试需使用 make_sample 构造样本，因此导入中新增了 make_sample。

关键文件：

- miles/ray/rollout/metrics.py（模块 rollout 指标；类别 source；类型 core-logic；符号 _compute_training_sample_metrics）：核心实现文件，新增 _compute_training_sample_metrics 函数并在 _compute_metrics_from_samples 中调用，是本次功能的主路径。

- tests/fast/ray/rollout/test_metrics.py（模块 测试；类别 test；类型 test-coverage；符号 TestTrainingSampleMetrics, test_compacted_rollouts_are_counted_as_samples_but_rewarded_as_episodes, test_different_sibling_rewards_are_averaged_within_rollout_first, test_rollout_ids_are_scoped_by_prompt_group）：新增测试类覆盖新函数的核心行为，并增强既有测试以验证日志输出新指标，确保功能正确性。

关键符号：_compute_training_sample_metrics

关键源码片段

miles/ray/rollout/metrics.py

核心实现文件，新增 _compute_training_sample_metrics 函数并在 _compute_metrics_from_samples 中调用，是本次功能的主路径。

```
# miles/ray/rollout/metrics.py

def _compute_training_sample_metrics(args: Any, samples: list[Sample]) -> dict[str, float | int]:
    """统计训练样本数，并按 rollout 等权平均原始奖励。

    Session compaction 会把一个 rollout 拆成多个训练样本，导致原有 raw-reward
    指标按样本加权，长 rollout 权重过高。本函数先对同一 rollout 内的样本
    奖励求平均，再对所有 rollout 平均，使每个 rollout 的权重相等。
    """
    # key 使用（类型，group_index, 具体值）三元组，确保不同 prompt group 的
    # 相同 rollout_id 或 index 不会互相混淆。
    rewards_by_rollout: dict[tuple[str, int | None, int], list[float]] = {}
    # 若第一个样本 metadata 带 raw_reward，则全部使用 metadata 中的值；
    # 否则回退到样本自身计算的奖励值。
    use_metadata_reward = bool(samples and samples[0].metadata and "raw_reward" in
    samples[0].metadata)
    for position, sample in enumerate(samples):
        # 优先用 rollout_id 作为分组依据；缺失时回退到样本 index；
        # 最后回退到枚举位置，保证每个样本都有唯一归属。
        if sample.rollout_id is not None:
            rollout_key = ("rollout", sample.group_index, sample.rollout_id)
        elif sample.index is not None:
            rollout_key = ("sample", sample.group_index, sample.index)
        else:
            rollout_key = ("position", sample.group_index, position)

        raw_reward = sample.metadata["raw_reward"] if use_metadata_reward else sample.get_
        reward_value(args)
        rewards_by_rollout.setdefault(rollout_key, []).append(raw_reward)

    # 先求每个 rollout 的平均奖励，再对所有 rollout 取平均，实现等权。
    rollout_rewards = [sum(rewards) / len(rewards) for rewards in rewards_by_rollout.values()]
    return {
        "num_training_samples": len(samples),
        "episode_raw_reward": sum(rollout_rewards) / len(rollout_rewards) if rollout_rewards else
```

```
        0.0,
    }
```

tests/fast/ray/rollout/test_metrics.py

新增测试类覆盖新函数的核心行为，并增强既有测试以验证日志输出新指标，确保功能正确性。

```
# tests/fast/ray/rollout/test_metrics.py

class TestTrainingSampleMetrics:
    def test_compacted_rollouts_are_counted_as_samples_but_rewarded_as_episodes(self):
        # compaction 把一个 rollout 拆成 3 个样本，另一个 rollout 只有 1 个样本
        # 样本总数应为 4，但奖励需按 rollout 等权平均，因此两个 rollout 各占一半
        args = make_args(reward_key=None)
        samples = [
            make_sample(index=0, rollout_id=10, reward=1.0),
            make_sample(index=0, rollout_id=10, reward=1.0),
            make_sample(index=0, rollout_id=10, reward=1.0),
            make_sample(index=1, rollout_id=11, reward=0.0),
        ]
        out = _compute_training_sample_metrics(args, samples)
        assert out["num_training_samples"] == 4
        assert out["episode_raw_reward"] == pytest.approx(0.5)

    def test_different_sibling_rewards_are_averaged_within_rollout_first(self):
        # 同一 rollout 内奖励不同，先组内平均，再等权平均
        args = make_args(reward_key=None)
        samples = [
            make_sample(index=0, rollout_id=10, reward=0.0),
            make_sample(index=0, rollout_id=10, reward=1.0),
            make_sample(index=1, rollout_id=11, reward=1.0),
        ]
        out = _compute_training_sample_metrics(args, samples)
        assert out["episode_raw_reward"] == pytest.approx(0.75)

    def test_rollout_ids_are_scoped_by_prompt_group(self):
        # 不同 prompt group 的 rollout_id 可能相同，必须按 group_index 隔离
        args = make_args(reward_key=None)
        samples = [
            make_sample(group_index=0, index=0, rollout_id=10, reward=1.0),
            make_sample(group_index=0, index=0, rollout_id=10, reward=1.0),
            make_sample(group_index=1, index=1, rollout_id=10, reward=0.0),
        ]
        out = _compute_training_sample_metrics(args, samples)
        assert out["episode_raw_reward"] == pytest.approx(0.5)
```

评论区精华

该 PR 仅有 bot 提示评论和两位 reviewer 的批准，未产生实质性 review 讨论。主要关注点隐含在测试用例中：如何在 compaction 场景下正确按 rollout 分组、如何处理 metadata

raw_reward 与样本奖励的回退、以及 rollout ID 需按 group_index 作用域化以避免不同分组冲突。

- 暂无高价值评论线程

风险与影响

- 风险：新增指标在主 rollout 指标计算路径中执行，但仅在 _compute_metrics_from_samples 内新增一行调用，且函数本身无副作用，风险主要在于： 1) 若样本中 rollout_id 缺失但 index 存在且不唯一（如不同分组出现相同 index），会通过 group_index 区分，但若 group_index 为 None 则可能错误合并； 2) use_metadata_reward 仅在第一个样本决定，若后续样本元数据缺失或存在混合情况，可能导致 reward 取值不一致； 3) 空样本时返回 0.0，避免除零。整体风险较低。
  - 影响：影响范围局限于 rollout 指标收集与 W&B 可视化，不涉及训练核心逻辑。新增指标可帮助监控训练过程中实际样本数，并修正因 compaction 导致的奖励权重偏差，对使用 session compaction 的训练任务有直接收益。对未启用 compaction 的训练，新增指标不会改变现有指标数值，但会新增额外指标，可能影响 W&B 面板布局或下游消费指标的脚本。
  - 风险标记：新增指标不影响核心训练逻辑，涉及样本奖励计算与控制流，测试覆盖充分

关联脉络

- PR #2200 [RL] Add sampling-support log-prob primitives: 同属 PPO/rollout 指标与采样逻辑，为 on-policy 归一化铺路，与本次指标增强同属训练可观测性方向。
- PR #2716 fix: align MLA RoPE types with model configs: 同为模型配置与指标相关，但关联较弱，仅体现团队对训练细节的持续打磨。