PR #7037 完整报告

verl-project/verl

[trainer, ckpt] feat: support streaming dataloader and async trainer checkpoint recovery

合并时间：2026-07-14 19:44

原文链接：http://prhub.com.cn/verl-project/verl/pull/7037

执行摘要

- 一句话：支持流式数据加载与异步训练器检查点恢复
- 推荐动作：建议团队精读本 PR，特别是 _reissue_inflight_prompts、_add_batch_to_generate 和 ReplayBuffer 的 drop/refill 实现，是异步训练器检查点恢复的核心。注意 review 中已识别的忙等待和共享引用问题需确认已修复（最终合并版本可能已包含）。

功能与动机

PR body 明确指出动机：支持从 dataloader 添加任意数量的 prompt 到生成，而不受 train_batch_size 限制（"supports adding arbitrary number of prompts from dataloader to generation without the train_batch_size limitation"）。同时改进异步训练器的 drop 策略，允许在检查点恢复后重启待定 / 运行中的 prompt 并保留已完成样本。

实现拆解

1. 配置解耦与流式 fetching：在训练器配置中新增 gen_batch_size 字段，与原有的 train_batch_size 分离。_next_train_batch 方法根据 num_prompts 参数以 gen_batch_size 为单位从 dataloader 分块 fetch，然后将结果 coalesce 为一个批次提交给 rollout。_add_batch_to_generate 被重构为同时支持一次提交指定数量的 prompt，并返回实际提交数。
2. Drop 与 Refill 策略：ReplayBuffer 新增 refill_fn 构造函数参数，由训练器注入 _add_prompts_to_generate 方法。当采用 drop 策略且检测到陈旧 finished prompt 时，sample 方法会调用 _drop_stale_finished 删除超过 max_off_policy_threshold 步数的 prompt 组，并通过 refill_fn 补充等量的新 prompt。同时新增 _sampleable_terminal_keys 方法来过滤仅返回未过期的 terminal keys。
3. TransferQueue 检查点持久化：在 _save_checkpoint 和 _load_checkpoint 中分别调用 tq.save_checkpoint 和 tq.load_checkpoint 来持久化 TransferQueue 中所有轨迹和 prompt 状态。新增模块级函数 _tq_supports_checkpoint 用于检查 TransferQueue 版本（>=0.1.9）并验证 save_checkpoint/load_checkpoint API 可用。恢复时，fit 方法开始处会调用 _reissue_inflight_prompts，将队列中状态为 pending 或 running 的 prompt 重新提交给 agent_loop_manager，同时清理这些 prompt 下已有的 session 轨迹（因部分生成不可恢复）。

4. 配置与工程配套：更新了所有生成的训练器 YAML 配置文件（_generated_*.yaml）和 legacy_data.yaml 以包含 gen_batch_size 字段。对 rollout_skip.py 中的 SkipManager 进行了调整，使其在 gen_batch_size 为 None 时正确回退到 train_batch_size。
5. 测试覆盖：新增了 test_streaming_feed_on_cpu.py（覆盖 num_prompts 解析逻辑）、test_reissue_inflight_on_cpu.py（覆盖检查点恢复的 re-issue 流程和 tq.save_checkpoint/load_checkpoint 的 round-trip）、test_replay_buffer_on_cpu.py（覆盖 drop 策略的 stale 过滤和 refill 行为）。同时全局 CI 测试也会覆盖这些路径。

关键文件：

- verl/trainer/ppo/v1/trainer_base.py（模块 训练器；类别 source；类型 dependency-wiring；符号 _tq_supports_checkpoint, _reissue_inflight_prompts, _add_batch_to_generate, _next_train_batch）：核心训练器文件，变更包括流式加载、drop/refill、检查点恢复的关键入口和主要逻辑。
- verl/trainer/ppo/v1/replay_buffer.py（模块 重放缓存；类别 source；类型 core-logic；符号 _accumulate_drop_metrics, _sampleable_terminal_keys, _drop_max_off_policy_samples, _drop_stale_finished）：重放缓存核心逻辑，新增 drop 筛选和 refill 回调，实现了丢弃陈旧 prompt 并自动补充。
- tests/trainer/ppo/v1/test_reissue_inflight_on_cpu.py（模块 恢复逻辑测试；类别 test；类型 test-coverage；符号 tq_init, _force_tq_checkpoint_supported, partition_id, test_tq_checkpoint_guard_checks_version_and_api_capabilities）：新增 CPU 单元测试，覆盖检查点恢复的 re-issue 逻辑和 TransferQueue 的 save/load round-trip。
- tests/trainer/ppo/v1/test_streaming_feed_on_cpu.py（模块 流式加载测试；类别 test；类型 test-coverage；符号 _resolve_num_prompts, _num_fetches, _num_submissions, _steps_per_epoch）：新增 CPU 单元测试，覆盖流式加载的 num_prompts 解析逻辑和 steps_per_epoch 计算。
- tests/trainer/ppo/v1/test_replay_buffer_on_cpu.py（模块 重放缓存测试；类别 test；类型 test-coverage；符号 FakeRefiller, init, call, test_has_enough_samples_drop_counts_only_fresh_terminal）：扩展现有重放缓存测试，覆盖 drop 策略的 stale 过滤和 refill 行为。
- verl/utils/skip/rollout_skip.py（模块 跳过管理器；类别 source；类型 core-logic）：SkipManager 的 gen_batch_size 解析逻辑调整，使其在 gen_batch_size 为 None 时回退到 train_batch_size。
- verl/trainer/config/data/legacy_data.yaml（模块 数据配置；类别 config；类型 configuration）：数据配置文件中新增 gen_batch_size 字段，是流式加载的配置入口。

关键符号：_tq_supports_checkpoint, _reissue_inflight_prompts, _add_batch_to_generate, _next_train_batch, _accumulate_drop_metrics, _sampleable_terminal_keys, _drop_stale_finished, _add_prompts_to_generate

关键源码片段

verl/trainer/ppo/v1/trainer_base.py

核心训练器文件，变更包括流式加载、drop/refill、检查点恢复的关键入口和主要逻辑。

```
# 检查 TransferQueue 是否支持检查点功能（版本 >= 0.1.9 且 API 可调用）
def _tq_supports_checkpoint() -> bool:
    try:
        version_supported = Version(getattr(tq, "__version__", "")) >= Version("0.1.9")
    except InvalidVersion:
        return False
    return (
        version_supported
        # 确保 save_checkpoint 和 load_checkpoint 是可调用的
        and callable(getattr(tq, "save_checkpoint", None))
        and callable(getattr(tq, "load_checkpoint", None))
    )
```

verl/trainer/ppo/v1/replay_buffer.py

重放缓存核心逻辑，新增 drop 筛选和 refill 回调，实现了丢弃陈旧 prompt 并自动补充。

```
# ---------- 新函数：合并丢弃指标 ----------
def _accumulate_drop_metrics(acc: dict, new: dict, dropped: int) -> None:
    """Merge one poll iteration's drop metrics into ``acc`` in place."""
    count_key = next((k for k in new if k.endswith("/dropped_samples")), None)
    prev_total = acc.get(count_key, 0) if count_key else 0

    for key, value in new.items():
        if key.endswith("/dropped_samples"):
            # 累加丢弃样本数量
            acc[key] = acc.get(key, 0) + value
        elif key.endswith("/dropped_samples_staleness/mean"):
            # 加权平均更新平均值
            denom = prev_total + dropped
            acc[key] = (acc.get(key, 0.0) * prev_total + value * dropped) / denom if denom else
            value
        elif key.endswith("/dropped_samples_staleness/max"):
            acc[key] = max(acc.get(key, value), value)
        elif key.endswith("/dropped_samples_staleness/min"):
            acc[key] = min(acc.get(key, value), value)

# ---------- ReplayBuffer 初始化片段（新增 refill_fn 参数） ----------
class ReplayBuffer:
    def __init__(
        self,
        trainer_mode: str,
        trainer_config: DictConfig,
        max_off_policy_threshold: int,
        max_off_policy_strategy: str,
        sampler_kwargs: DictConfig,
        poll_interval: float = 2.0,
        refill_fn=None, # 新增参数：用于丢弃后补充新 prompt 的回调
    ):
        # 其他原有初始化属性 ...
```

```
        self.refill_fn = refill_fn # 存储 refill 回调引用
```

评论区精华

核心讨论集中在以下方面：

- 无限忙等待风险：gemini-code-assist[bot] 指出在 ReplayBuffer.sample 的 drop 循环中，如果 dropped > 0 后立即 continue 并再次同步元数据，可能因 TransferQueue 最终一致性导致 tight loop。建议添加 time.sleep(poll_interval)。
- 列表乘字典共享引用 Bug：同一机器人发现 [{...}] * n 会导致多个元素引用同一字典，后续修改会互相影响。建议改用列表解析。
- _add_prompts_to_generate 设计：wuxibin89 询问为何不直接复用 _add_batch_to_generate，Begunner 解释是为了避免修改 SkipManager 并保持语义清晰。
- 检查点恢复范围：wuxibin89 指出 sync 模式也可使用检查点，Begunner 认为 sync 模式没有 inflight 样本，无需保存，但最终可能仍需要支持。
- 代码组织建议：wuxibin89 建议将 TransferQueue save/load 提取为独立函数，Begunner 接受。
- 版本检查：wuxibin89 建议检查 TransferQueue 版本，当前已通过 _tq_supports_checkpoint 实现。
  - ReplayBuffer sample drop 循环可能无限忙等待 (correctness): PR 合并前可能已通过其他 commit 修复（如 'fix drop, sample will use the same snapshot of tq'），但无明确确认。
  - 列表乘字典导致共享引用 (correctness): 后续 commit 可能已修复，PR 合并时未再提出。
  - 为何引入 _add_prompts_to_generate 而非复用 _add_batch_to_generate (design): 接受该设计。
  - Sync 模式是否也应支持检查点保存 / 恢复 (design): 当前仅针对 async 模式，sync 模式未启用。未进一步修改。
  - 将 TransferQueue save/load 提取为独立函数 (style): 接受建议，预计后续提交中已提取。
  - 检查 TransferQueue 版本 (design): 建议已实现，Begunner 回复 'Get'。

风险与影响

- 风险：
  - 检查点兼容性：如果未来的 TransferQueue 修改了序列化格式，会导致 checkpoint 不兼容。当前仅通过版本号 >=0.1.9 检查，未考虑降级场景。
  - Drop 策略的忙等待风险：如 review 中所指，在 sample 循环的 drop 分支中缺少 time.sleep，可能导致 CPU 忙等。
  - 依赖版本约束：_tq_supports_checkpoint 要求 TransferQueue >=0.1.9，用户需要手动升级，否则降级为不支持 checkpoint fallback。
- 影响：
  - 用户影响：V1 PPO 异步训练器用户可通过设置 data.gen_batch_size=1 启用流式模式，并受益于检查点恢复功能。

- 系统影响：新增对 TransferQueue >=0.1.9 的版本依赖；checkpoint 文件增加 TransferQueue 快照；训练步骤中添加了 feed 耗时统计。
- 团队影响：需要确保 CI 环境中 TransferQueue 版本满足要求；新增的单元测试覆盖了关键路径，降低回归风险。
- 风险标记：检查点兼容性，Drop 循环可能忙等待，TransferQueue 版本依赖

关联脉络

- PR #6897 [tool, rollout] feat: Adapt SkipManager on Trainer V1: 修改了相同的 trainer_base.py 和 replay_buffer.py，是 SkipManager 适配的前置工作。
- PR #7032 [tool, rollout] feat: support parameter sync steps in Skip Manager: 修改了相同的 rollout_skip.py，本 PR 中的 gen_batch_size 回退逻辑与之相关。