PR #24070 完整报告

sgl-project/sglang

[BugFix] Fix rid_to_state leak for aborted queued requests

合并时间：2026-05-20 16:32

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24070

执行摘要

- 一句话：修复队列 abort 后 rid_to_state 泄漏导致重复请求 ID 错误
- 推荐动作：值得精读。这是一个典型的资源泄漏 bug 修复，代码简洁但影响明确。特别关注测试设计：使用 __new__ 绕过 __init__ 的 mock 方式，以及利用 frozenset 分类字段避免硬编码，都是良好的实践。

功能与动机

在 RL 训练工作流中，权重更新或 rollout 同步会频繁触发全局请求 abort 和 resubmit。队列中的请求在 abort 后，其 rid_to_state 条目未被清理，导致下一次 resubmit 时因为检测到重复 ID 而失败。PR body 中详细分析了 running 请求和 queued 请求的不同 abort 路径，指出 queued 路径缺少清理。

实现拆解

1. 定位问题：在 tokenizer_manager.py 的 _handle_abort_req 方法中发现缺少 del self.rid_to_state[rid] 调用。
2. 修复：在 _handle_abort_req 结尾处，向 out 字典赋值后，立即执行 del self.rid_to_state[recv_obj.rid]，与 _handle_batch_output 中 finished 请求的处理一致。
3. 测试：新增 test/registered/unit/managers/test_tokenizer_manager_rid_cleanup.py，通过 mock TokenizerManager 避免加载模型，直接测试三个代码路径：abort 清理、batch 输出清理、重复请求 ID 检测。测试注册为 CPU 测试 (est_time=15)，CI 执行。

关键文件：

- python/sglang/srt/managers/tokenizer_manager.py（模块 状态管理；类别 source；类型 core-logic；符号 _handle_abort_req）：核心修复文件，在 _handle_abort_req 中添加了 del self.rid_to_state[recv_obj.rid]，修复了队列 abort 请求后状态泄漏的问题。
- test/registered/unit/managers/test_tokenizer_manager_rid_cleanup.py（模块 测试；类别 test；类型 test-coverage；符号 _make_tokenizer_manager, _make_req_state, _make_abort_req, _make_batch_str_output）：新增的单元测试文件，通过 mock TokenizerManager 验证了 abort、正常完成和重复 ID 检测三个核心场景，确保修复正确性并防止回归。

关键符号：_handle_abort_req, _make_tokenizer_manager, _make_req_state, _make_abort_req, _make_batch_str_output, test_abort_removes_rid_from_state,

test_abort_allows_resubmit_same_rid, test_abort_sets_finished_and_notifies

关键源码片段

python/sglang/srt/managers/tokenizer_manager.py

核心修复文件，在 _handle_abort_req 中添加了 del self.rid_to_state[recv_obj.rid]，修复了队列 abort 请求后状态泄漏的问题。

```
# python/sglang/srt/managers/tokenizer_manager.py
# 在 _handle_abort_req 方法结尾处添加清理逻辑，确保 rid_to_state 无残留

        # ... 前面构建 out 字典的代码 ...
        out = {
            "text": state.get_text(),
            "output_ids": output_ids,
            "meta_info": meta_info,
        }
        # 关键修复：立即从 rid_to_state 中删除该请求 ID，
        # 与 _handle_batch_output 中已完成请求的处理保持一致
        del self.rid_to_state[recv_obj.rid]

        state.out_list.append(out)
        state.event.set()
```

test/registered/unit/managers/test_tokenizer_manager_rid_cleanup.py

新增的单元测试文件，通过 mock TokenizerManager 验证了 abort、正常完成和重复 ID 检测三个核心场景，确保修复正确性并防止回归。

```
# test/registered/unit/managers/test_tokenizer_manager_rid_cleanup.py
# 单元测试示例：验证 abort 后 rid 被清理且允许重新提交

    def test_abort_removes_rid_from_state(self):
        """Abort 后 rid 应从 rid_to_state 中移除"""
        tm = _make_tokenizer_manager()
        rid = "abort_test_rid"
        # 模拟添加一个请求状态
        tm.rid_to_state[rid] = _make_req_state(rid)
        # 构造 AbortReq 并调用处理函数
        abort_req = _make_abort_req(rid)
        tm._handle_abort_req(abort_req)
        # 验证 rid 已被移除
        self.assertNotIn(rid, tm.rid_to_state)

    def test_abort_allows_resubmit_same_rid(self):
        """Abort 清理后，相同 rid 可以重新提交"""
        tm = _make_tokenizer_manager()
        rid = "resubmit_after_abort_rid"
        # 第一次提交
        tm.rid_to_state[rid] = _make_req_state(rid)
```

```
# 模拟 abort
tm._handle_abort_req(_make_abort_req(rid))
# 第二次提交应成功，不会抛出异常
tm._init_req_state(rid, _make_req_state(rid).obj)
self.assertIn(rid, tm.rid_to_state)
```

评论区精华

- 审核者 @hnyls2002 建议不要添加端到端测试，而应使用 mock 单元测试，以免引入不必要的复杂性。
- @alexnails 询问 E2E 测试中是否可以使用 threading 保证顺序，并质疑一些代码的必要性。
- @guoyuhong 采纳意见，将初版 E2E 测试替换为 mock 单元测试，并解释新测试通过 __new__ 绕过 __init__ 避免模型加载，10 个测试全部在 CPU 上运行。
- 最终 @alexnails 批准了修改。
- 测试策略：单元测试 vs E2E 测试 (testing): @guoyuhong 将初版的 E2E 测试完全替换为 mock 单元测试，并说明新测试通过 __new__ 绕过模型加载，所有测试在 CPU 上运行。
- E2E 测试实现细节 (design): 该 E2E 测试被替换，问题论点无关。
- 测试代码简洁性 (style): 最终移除了整个 E2E 测试文件，相关问题不再存在。

风险与影响

- 风险：极低风险。修复仅增加了一行删除字典项的代码，且该操作在其他路径中已经存在，逻辑一致。新增的单元测试覆盖了 abort 和正常完成路径，能够防止回归。该变更不会影响正常推理路径 (running 请求的 abort 路径不变)。
- 影响：对用户：修复了 RL 训练中 resubmit 失败的问题，提高了稳定性。对系统：无性能影响，仅增加了单行 O(1) 操作。对团队：新增的测试框架可作为后续 TokenizerManager 相关测试的参考。
- 风险标记：状态管理遗缺，单点修复，测试覆盖新增

关联脉络

- 暂无明显关联 PR