

PR #56249 完整报告

vllm-project/vllm

[Bugfix][Frontend] Handle aborted requests in beam search

合并时间: 2026-09-15 16:17

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/56249>

执行摘要

- 一句话: 修复 beam search 中止崩溃并保留部分输出
- 推荐动作: 值得精读, 优先级中等: 改动约 170 行、3 个源文件 + 2 个测试文件, 阅读成本低但涉及三个可复用的设计决策。建议重点关注 (1) online.py 用 break + 全量标记 abort 而非 return, 为什么这样才能保留部分输出; (2) offline.py 用 instance_start_and_end 切片实现 prompt 级隔离, 以及把未完成 beam 放进 instance.completed 的后果; (3) finish_reason="abort" 成为对外可见契约, 以及 get_beam_search_score 中 max(seq_len, 1) 这类边界防护对排序路径的普适价值。后续可跟进: 补一条真实 /pause?mode=abort 的端到端测试, 并在 API 文档中说明 abort 的返回语义。

功能与动机

PR body 明确描述了触发条件: Aborting an inner beam-search request before it generates a token can return logprobs=[]. Both the online and offline paths only check for None before reading logprobs[0], so this raises IndexError. With /pause?mode=abort, a running /v1/completions request returns HTTP 500. 作者在单卡 A10 + Qwen/Qwen3.5-0.8B 上复现: 中止前三次请求全部 HTTP 500 (list index out of range), 中止后三次均返回 HTTP 200 且带 8 个标记为 abort 的 choice。除崩溃本身外, 还有两个衍生目标: 一次性中止所有活跃 beam, 避免向已暂停的引擎继续提交下一轮子请求; 以及在离线批量中让被中止 prompt 之外的请求继续生成。另外, 被保留的 beam 可能只剩一个 EOS prompt token, 评分时会除以 0, 需要一并防护。

实现拆解

1. 崩溃定位 (在线入口): vllm/entrypoints/generate/beam_search/online.py 的 BeamSearchOnlineMixin.beam_search 每轮为每个 beam 生成独立子请求 (request_id_batch = f"{request_id}-{random_uuid()}", 子请求 ID 为 -beam-{i}), 收齐结果后直接在同一个循环里读取 result.outputs[0].logprobs[0]。原来的守卫只判 logprobs is not None, 而被中止的子请求会返回空 list, 导致 IndexError → HTTP 500。
2. 在线路径: 先判定终止原因, 再收集候选: 把原先混在一起的候选收集循环拆成两遍。第一遍只做终止判断——先保持原有 finish_reason == "error" 行为 (yield 错误 RequestOutput 并 return), 随后新增 any(result.outputs[0].finish_reason == "abort" for result in output) 分支: 命中即把 all_beams 全部标记 finish_reason="abort" 并 break 主循环。用 break 而非 return, 使函数走到尾部统一产出, 从而保留每个 beam 在

最后一个完成步骤的 `tokens`、`logprobs` 与 `cum_logprob`；同时因为已经跳出 `for _ in range(max_tokens)`，不会再向已暂停的引擎提交下一轮子请求。`candidates = []` 及其打分循环整体下移到 `abort` 检查之后，从根上消除对空 `logprobs` 的索引。

3. 离线路径：按 `instance` 做 `prompt` 级隔离：`vllm/entrypoints/generate/beam_search/offline.py` 的 `_beam_search_step` 以 `for (start, end), instance in zip(instance_start_and_end, instances_batch)` 逐 `prompt` 处理。新增 `instance_output = output[start:end]` 切片判断，若该 `prompt` 的任一子结果 `finish_reason == "abort"`，就把这一段 `all_beams` 逐个标记 `abort` 并 `append` 进 `instance.completed` (`tokens` 与分数停在最后一个完成 `step`)，清空 `instance.beams` 后 `continue`。由于 `instance.completed` 是最终结果的来源，被中止的 `prompt` 仍能返回部分序列，而其他 `instance` 完全不受影响，继续走原有的 `allowed_sets` 过滤、EOS 归并、排序与 `beam_width` 截断。改动同时删除了 `# if logprobs is None, the sequence completed due to max-model-len or abortion`。这条已失效的注释——`abortion` 现在有独立分支。
4. 打分函数除零防护：`vllm/entrypoints/generate/beam_search/utils.py` 的 `get_beam_search_score` 中，`tokens[-1] == eos_token_id` 时会 `seq_len -= 1`；`aborted beam` 可能只保留一个 EOS `prompt token`，使 `seq_len == 0`，`0 ** length_penalty` 作为分母抛 `ZeroDivisionError`。改为 `max(seq_len, 1) ** length_penalty`。该函数被 `create_sort_beams_key_function` 包装为排序 `key`，是 `beam` 选择与最终输出共用的公共路径，因此这一行影响面超出 `abort` 场景。
5. 测试与验证配套：`tests/samplers/test_beam_search.py` 新增 `test_beam_search_abort_returns_partial_outputs_and_continues_other_prompts`，用 `LLM.__new__(LLM) + Mock()` 组装轻量 LLM，并 `monkeypatch _preprocess_cmpl` 与 `_render_and_run_requests` 伪造引擎返回；`tests/samplers/test_beam_search_online.py` 新增 `test_beam_search_abort_returns_partial_outputs`，`monkeypatch serving.engine_client.generate` 并用 `calls` 计数断言 `abort` 后不再提交下一步。两者都参数化 `abort_after ∈ {0, 1}`、`prompt_token ∈ {1, 0}`、`terminal_logprobs ∈ {None, [], [{11: Logprob(-0.1)}]}`，额外覆盖 EOS-only `prompt`、混合 `aborted/normal children` 与独立离线 `prompt`。PR body 记录修复前 18 个新用例全失败，修复后连同 2 个既有 `skip_special_tokens` 用例共 21 `passed`，并额外人工跑了 `/pause?mode=abort` 的在线与离线冒烟。

关键文件：

- `vllm/entrypoints/generate/beam_search/online.py`（模块 集束搜索；类别 `source`；类型 `core-logic`；符号 `BeamSearchOnlineMixin.beam_search`）：在线 `beam search` 主循环所在地，也是 HTTP 500 的直接触发点：新增 `abort` 检查并重排候选收集循环，是本次修复的核心。
- `vllm/entrypoints/generate/beam_search/offline.py`（模块 集束搜索；类别 `source`；类型 `core-logic`；符号 `BeamSearchOfflineMixin._beam_search_step`）：离线批量路径的每步推进函数，新增按 `instance`（即 `prompt`）切片的 `abort` 判断，实现 " 被中止 `prompt` 返回部分结果、其他 `prompt` 继续 " 的隔离语义。
- `vllm/entrypoints/generate/beam_search/utils.py`（模块 集束搜索；类别 `source`；类型 `core-logic`；符号 `get_beam_search_score`）：`aborted beam` 只保留一个 EOS `prompt`

token 时，评分函数会除以 0；这里加的下界保护同时被排序 key 复用，属公共路径的最小必要修改。

- tests/samplers/test_beam_search.py (模块 集束搜索；类别 test；类型 test-coverage；符号 test_beam_search_abort_returns_partial_outputs_and_continues_other_prompts, run_requests)：离线路径的回归测试，覆盖中止后保留部分输出、EOS-only prompt、以及同批次其他 prompt 继续生成；也是 review 中讨论 " 行为契约 " 的落点。
- tests/samplers/test_beam_search_online.py (模块 集束搜索；类别 test；类型 test-coverage；符号 test_beam_search_abort_returns_partial_outputs, generate)：在线路径的回归测试，用 calls 计数断言 abort 后不再提交下一步，并覆盖 None / 空 / 非空三种 terminal logprobs；用例命名在 review 中被要求澄清。

关键符号：BeamSearchOnlineMixin.beam_search,
BeamSearchOfflineMixin._beam_search_step, get_beam_search_score,
create_sort_beams_key_function, test_beam_search_abort_returns_partial_outputs,
test_beam_search_abort_returns_partial_outputs_and_continues_other_prompts

关键源码片段

vllm/entrypoints/generate/beam_search/online.py

在线 beam search 主循环所在地，也是 HTTP 500 的直接触发点：新增 abort 检查并重排候选收集循环，是本次修复的核心。

```
# vllm/entrypoints/generate/beam_search/online.py
# 每轮把所有 beam 作为独立子请求并发提交，子请求 ID 形如
# f"{request_id}-{random_uuid()}-beam-{i}"; /pause?mode=abort
# 会中止这一批子请求，并让它们的 logprobs 变成空 list 而不是 None
output = [x[0] for x in await asyncio.gather(*tasks)]

# 第一遍：只检查终止类 finish_reason，完全不碰 logprobs
for result in output:
    if result.outputs[0].finish_reason == "error":
        yield RequestOutput(
            request_id=request_id,
            prompt=prompt_text,
            outputs=[
                CompletionOutput(
                    index=0,
                    text="",
                    token_ids=[],
                    cumulative_logprob=None,
                    logprobs=None,
                    finish_reason="error",
                )
            ],
            finished=True,
            prompt_token_ids=prompt_token_ids,
            prompt_logprobs=None,
```

```

    )
    return

# 新增核心分支：任一子请求被 abort，就整体停止本轮及后续解码。
# 这里用 break 而不是 return，让函数走到尾部统一产出结果，
# 从而保留每个 beam 在最后一个完成步骤的 tokens / logprobs / cum_logprob；
# 同时跳出 for _ in range(max_tokens)，不再向已暂停的引擎提交下一轮请求
if any(result.outputs[0].finish_reason == "abort" for result in output):
    for beam in all_beams:
        beam.finish_reason = "abort"
    break

# 候选收集被下移到 abort 检查之后：被中止的 beam 永远不会再执行
# result.outputs[0].logprobs[0]，从根上消除 IndexError 与 HTTP 500
candidates = []
for i, result in enumerate(output):
    current_beam = all_beams[i]

    if result.outputs[0].logprobs is not None:
        logprobs = result.outputs[0].logprobs[0]
        # ... 后续 EOS / 非 EOS 候选打分、排序截断逻辑保持不变

```

vllm/entrypoints/generate/beam_search/offline.py

离线批量路径的每步推进函数，新增按 instance（即 prompt）切片的 abort 判断，实现 " 被中止 prompt 返回部分结果、其他 prompt 继续 " 的隔离语义。

```

# vllm/entrypoints/generate/beam_search/offline.py
for (start, end), instance in zip(instance_start_and_end, instances_batch):
    # 新增：按 prompt（instance）切片判断中止，做到 prompt 级隔离——
    # 一个 prompt 被 abort 不影响同一离线批次里的其他 prompt
    instance_output = output[start:end]
    if any(
        result is not None and result.outputs[0].finish_reason == "abort"
        for result in instance_output
    ):
        # 把该 prompt 当前的 beam 原样保留（tokens / logprobs / cum_logprob
        # 都停留在最后一个完成的 beam step），标记 abort 后放进 completed，
        # 这些部分序列会随最终结果一起返回给调用方
        for beam, result in zip(all_beams[start:end], instance_output):
            if result is not None:
                beam.finish_reason = "abort"
                instance.completed.append(beam)
        # 清空活跃 beam 并 continue：该 prompt 不再提交下一步，
        # 也不会再走到下面 logprobs[0] 的索引逻辑
        instance.beams = []
        continue

    instance_new_beams = []
    for i in range(start, end):

```

```

current_beam = all_beams[i]
result = output[i]

if result is None:
    continue

if result.outputs[0].logprobs is not None:
    logprobs = result.outputs[0].logprobs[0]
    # ... 原有的 allowed_sets 语法过滤、EOS 归并、
    # sort_beams_key 排序与 beam_width 截断逻辑保持不变

```

vllm/entrypoints/generate/beam_search/utils.py

aborted beam 只保留一个 EOS prompt token 时，评分函数会除以 0；这里加的下界保护同时被排序 key 复用，属公共路径的最小必要修改。

```

# vllm/entrypoints/generate/beam_search/utils.py
def get_beam_search_score(
    tokens: list[int],
    cumulative_logprob: float,
    eos_token_id: int,
    length_penalty: float = 1.0,
) -> float:
    """按 length penalty 归一化的 beam 分数，用于 beam 排序与最终输出。"""
    seq_len = len(tokens)
    if tokens[-1] == eos_token_id:
        # 以 EOS 结尾时不计入 EOS 本身，避免惩罚 " 提前结束 " 的序列
        seq_len -= 1

    # 被中止的 beam 可能只保留一个 EOS prompt token：此时 seq_len 会被减到 0，
    # 0 ** length_penalty 作为分母会抛 ZeroDivisionError，因此用 max(seq_len, 1) 兜底。
    # 该函数同时被 create_sort_beams_key_function 用作排序 key，属于公共路径
    return cumulative_logprob / (max(seq_len, 1) ** length_penalty)

```

评论区精华

讨论量不大，但有一条明确的命名争议：DarkLight1337 在 [tests/samplers/test_beam_search_online.py](#) 的新增用例上提问 What does the `preserves_prefixes` part of the function name refer to here?，作者解释 prefixes 指每个 beam 在被中止那一步之前已生成的 token，并接受反馈把用例改名为 `test_beam_search_abort_returns_partial_outputs`（离线侧对应 `.._and_continues_other_prompts`），改名落在独立的第二个 commit [Test] Clarify beam search abort test names 里。除此之外都是流程性往来：claude[bot] 提示 fork PR 默认关闭自动 review，需要维护者手动 [@claude review](#)；DarkLight1337 两次 `/ci run`，第二次之前 github-actions 提示分支落后上游 24 个 commit 需先 rebase，作者 merge main 后在 323b890 上触发 Buildkite #89036，最终 DarkLight1337 APPROVED 并留言 Thanks。整个 review 没有对 abort 语义本身提出异议，属于低争议、以测试可读性为主的讨论。

- 测试命名 `preserves_prefixes` 语义不清 (style): 作者解释 prefixes 指每个 beam 在被中止那一步之前已生成的 token，并接受反馈把用例改名为 `test_beam_search_abort_returns_`

partial_outputs, 同时把离线用例改名为 test_beam_search_abort_returns_partial_outputs_and_continues_other_prompts; 改名落在独立 commit [Test] Clarify beam search abort test names。

- fork PR 的 CI 触发与自动化 review 门禁 (other): 作者 merge main 后在 323b890 触发 Buildkite CI #89036, DarkLight1337 随后 APPROVED 并留言 Thanks 完成合并。

风险与影响

- 风险: 1) 中止语义被放大为 " 整轮取消 ": online.py 用的是 any(...finish_reason == "abort" ...), 只要一个子请求被中止, 整轮所有 beam 都被标记 abort 并 break。在 /pause?mode=abort 这类引擎整体暂停场景下是正确取舍, 但如果未来出现只中止部分子请求的情况, 其他 beam 已生成的内容会被一并丢弃。
- 2) completed 列表语义被扩展: offline.py 把未自然结束的 beam 直接 append 到 instance.completed, 该列表原本主要承载 EOS / 长度上限结束的序列。下游若对 completed 做排序或过滤, 需要容忍 logprobs 更短、cum_logprob 更小的 " 部分序列 "; max(seq_len, 1) 只解决除零, 并未解决 aborted beam 与正常 beam 的分数可比性问题, length_penalty 下二者的归一化口径不再一致。
- 3) 对外 API 契约扩大: finish_reason="abort" 会经 CompletionOutput 暴露给 OpenAI 兼容接口, 而该取值不属于 OpenAI 规范的 stop/length/content_filter/tool_calls 集合, 做严格校验的客户端或 SDK 可能报错; 新测试已把它固化为契约, 建议同步文档说明。
- 4) 测试层次偏单元: 两个新用例都靠 monkeypatch 伪造引擎输出, 真实 /pause?mode=abort 与 beam search 的交互只在 PR body 中手工冒烟 (A10 + Qwen3.5-0.8B), 未进入 CI 端到端覆盖; 引擎侧中止行为若再变化, 这些测试不会失败。
- 5) 未覆盖的组合路径: LoRA、session_id、trace 以及 structured output 下的 abort 未单独覆盖; offline 的 abort 分支位于 allowed_sets 过滤之前, 等于中止时跳过语法过滤, 这对 structured output 场景是可接受但未明说的行为。
- 6) 回归面: 普通采样路径未改动, 风险集中在 vllm/entrypoints/generate/beam_search/*, 其中 get_beam_search_score 被所有 beam 排序复用, max(seq_len, 1) 只在 seq_len 为 0 时改变结果, 理论回归面很小。- 影响: 用户侧: 启用 /pause?mode=abort、下发取消或断连时, beam search 请求不再返回 HTTP 500, 而是拿到带 finish_reason="abort" 的部分 beam 结果 (每个 beam 保留最后一个完成步骤的 token 与 logprobs); 离线 LLM.beam_search 批量调用中单个 prompt 被中止不再让整批抛 IndexError, 同批其他 prompt 正常完成。系统侧: 改动只在前端编排层, 不触及调度器、KV cache 与采样内核; 副作用收益是中止后不再向已暂停引擎提交下一步, 减少无效请求与日志噪声。团队侧: 为 beam search 建立了明确的中止契约 (保留最后完成 step、prompt 级隔离、对外暴露 abort 状态), 后续若要调整 abort 语义可在此基础上扩展; 同时该 PR 依赖 fork + 手动 /ci run 的流程, 提示维护者需关注分支落后上游带来的 CI 门禁成本。- 风险标记: 中止语义放大为整轮取消, 非标准 finish_reason 暴露给 API, completed 列表语义被扩展, 测试依赖 Mock 引擎, 缺少端到端 pause/abort 覆盖

关联脉络

- PR #47126 [Bugfix] Fix beam search candidate indexing when logprobs count varies: 同一模块的前置修复，处理 logprobs 数量变化导致的候选索引越界；PR#56249 的 body 明确指出该修复 leaves this abort path unchanged，两者合起来覆盖了 logprobs 形状不稳定的两类情况（数量变化 vs. 空列表），label 组合 (bug/frontend/ready) 也完全一致。
- PR #55602 [Bugfix] Avoid nested rerank cancellation handlers: 同属前端请求中止 / 取消处理的修复线：该 PR 删除嵌套取消处理器以避免断连后仍处理请求，与 PR#56249 处理 abort 后不再向已暂停引擎提交下一步的动机一致，反映入口侧中止语义是近期持续修整的方向。