

PR #56639 完整报告

vllm-project/vllm

[Pooling] Support prompt embeddings in MRV2 decoder pooling

合并时间: 2026-09-13 13:49

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/56639>

执行摘要

- 一句话: MRV2 池化支持 prompt_embeds, token id 断言下沉到任务级
- 推荐动作: 值得精读, 但不必花大量时间: diff 极小, 重点看两处设计决策——(1) 为什么把 prompt_token_ids 放宽为 list[int] | None 而不是在 model_runner 里为 embeds 请求加特例; (2) 为什么断言放在 if pooling_params.requires_token_ids: 分支内而不是 _get_pooling_metadata 里。这两点体现了“输入校验跟随能力声明、在真正使用点失败”的模式, 可直接迁移到其他 MRV2 迁移修复中。若你在维护池化任务或新增 pooler, 请额外关注 requires_token_ids 的声明是否与实际实现一致, 并建议补一条 MRV2 下 embeds 与 token 输入混合 batch 的回归用例, 把本次手工验证固化进 CI。

功能与动机

PR body 明确指出这是 MRV1 与 MRV2 之间的回归: MRV2 的 prompt-embedding 状态本就支持预计算嵌入, 但池化请求路径无条件断言 prompt_token_ids 存在, 于是“Decoder pooling requests that supplied prompt_embeds therefore crashed before inference, even though most pooling tasks do not need token IDs”。触发场景是可复现的: 开启 enable_prompt_embeds 后把 {prompt_embeds: torch.randn(4, hidden_size)} 与普通字符串放在同一批请求里, main 分支报 AssertionError 与 EngineDeadError, 而大多数池化任务 (embedding、rerank) 本身并不需要 token id。修复目标是在不改变既有行为的前提下, 让 token id 只在声明需要它的任务上被强制要求。

实现拆解

本 PR 只有 2 个文件、+2/-2, 却落在 MRV1 → MRV2 迁移的请求入队契约上, 可按三层理解。

1. 调用侧放宽契约 (vllm/v1/worker/gpu/model_runner.py): add_requests 遍历 scheduler_output.scheduled_new_reqs 时, 原本在 if self.pooling_runner is not None: 分支内先执行 assert new_req_data.prompt_token_ids is not None, 再调用 self.pooling_runner.add_request(req_id, req_index, new_req_data.pooling_params, new_req_data.prompt_token_ids)。该断言隐含“凡池化请求必有 token id”的假设, 与 MRV2 允许 prompt-embeds-only 输入的事实冲突, 于是请求在入队阶段就崩溃。本次直接删除该断言, 把判定权交给下游。
2. 消费侧按能力声明校验 (vllm/v1/worker/gpu/pool/pooling_runner.py): add_request 的 prompt_token_ids 参数由 list[int] 改为 list[int] | None, 并在 if pooling_params.requires_token_ids: 分支内新增 assert prompt_token_ids is not None。

这样“是否需要 token id”由任务自身的 `requires_token_ids` 声明驱动：token 分类、SPLADE 等任务在缺 token id 时仍快速失败，embedding / rerank 等任务允许 None 入队。

3. 下游兼容性确认：同文件的 `_get_pooling_metadata` 早已用 `if any(params.requires_token_ids for params in pooling_params)`：决定是否构造 `prompt_token_ids_cpu`，`remove_request` 也用 `self.prompt_token_ids.pop(req_index, None)` 做删除，因此 `self.prompt_token_ids` 字典缺项不会破坏取用逻辑——这正是断言可以安全下沉到 `requires_token_ids` 分支的前提。
4. 测试与配套改动：本次没有新增或修改测试文件（`changed_files` 均为源码）。验证依赖 PR 描述中的手工 reproducer（`runner_v2 True outputs 2 embedding_dims [1024, 1024]`）以及既有用例：`tests/v1/worker/test_prompt_embeds_state.py`（4 passed）、`VLLM_USE_V2_MODEL_RUNNER=1` 下的 `tests/models/language/pooling/test_all_pooling_plus_chunked_prefill.py`（3 passed）、`test_splade_sparse_pooler.py`（12 passed）、`test_jina_reranker_v3.py::test_offline`（1 passed），整体 29 tests passed；精度对比表显示 prompt embeds 与 token id 两条路径的最大绝对差为 0、余弦相似度为 1.0。

关键文件：

- `vllm/v1/worker/gpu/pool/pooling_runner.py`（模块 池化执行器；类别 source；类型 core-logic；符号 `add_request`）：池化请求的注册入口，本次真正的契约变更点：`prompt_token_ids` 放宽为可选，断言下沉到 `requires_token_ids` 分支，决定了哪些任务允许没有 token id。
- `vllm/v1/worker/gpu/model_runner.py`（模块 模型运行器；类别 source；类型 data-contract；符号 `add_requests`）：池化请求的统一入队点，本次删除了对 `prompt_token_ids` 的无条件断言；正是这行断言把 `prompt-embeds-only` 请求变成 `EngineDeadError`。

关键符号：`add_request`, `add_requests`

关键源码片段

`vllm/v1/worker/gpu/pool/pooling_runner.py`

池化请求的注册入口，本次真正的契约变更点：`prompt_token_ids` 放宽为可选，断言下沉到 `requires_token_ids` 分支，决定了哪些任务允许没有 token id。

```
# 文件: vllm/v1/worker/gpu/pool/pooling_runner.py
def add_request(
    self,
    req_id: str,
    req_index: int,
    pooling_params: PoolingParams,
    # 改动点 1: prompt_token_ids 由必填 list[int] 放宽为 Optional。
    # MRV2 允许请求只带预计算的 prompt_embeds (Qwen3-Embedding 等 decoder pooling),
    # 这类请求本就没有 token id, 属于合法输入而非错误。
    prompt_token_ids: list[int] | None,
) -> None:
    task = pooling_params.task
    if task not in self.supported_tasks:
```

```

    raise ValueError(
        f"Unsupported task: {task!r}. "
        f"Supported tasks: {sorted(self.supported_tasks)}"
    )
# 先让 pooler 按任务类型改写参数，再做后续登记，保证 pooling_params 已定型
self.model.pooler.get_pooling_updates(task).apply(pooling_params)
self.late_interaction_runner.register_request(req_id, pooling_params)
self.pooling_params[req_index] = pooling_params
self.pooling_states[req_index] = PoolingStates()

# 改动点 2：把 token id 的强制校验从调用方（model_runner）下移到真正使用它的分支。
# 只有声明 requires_token_ids 的任务（token 分类、SPLADE 等）才要求 token id 非空；
# embedding / rerank 等任务即使 prompt_token_ids 为 None 也能正常入队。
# 之所以安全：同文件 _get_pooling_metadata 用 any(params.requires_token_ids ...)
# 作为是否构造 prompt_token_ids_cpu 的开关，remove_request 也用 pop(req_index, None)
# 删除条目，因此下游不会读到未登记的 None。
if pooling_params.requires_token_ids:
    assert prompt_token_ids is not None
    self.prompt_token_ids[req_index] = torch.tensor(
        prompt_token_ids, dtype=torch.int64
    )

```

vllm/v1/worker/gpu/model_runner.py

池化请求的统一入队点，本次删除了对 prompt_token_ids 的无条件断言；正是这行断言把 prompt-embeds-only 请求变成 EngineDeadError。

```

# 文件：vllm/v1/worker/gpu/model_runner.py（保留本次改动附近的核心分支）
def add_requests(self, scheduler_output: SchedulerOutput) -> None:
    for new_req_data in scheduler_output.scheduled_new_reqs:
        assert new_req_data.prefill_token_ids is not None
        req_id = new_req_data.req_id

# 中间省略：流式输入更新先 _remove_request，再登记 req_states / req_index、
# 自适应投机校验状态等既有逻辑。

if self.pooling_runner is not None:
    assert new_req_data.pooling_params is not None
    # 本次删除了这一行：assert new_req_data.prompt_token_ids is not None
    # 原因是它把“是否需要 token id”的判断强加给所有池化请求，
    # 使 prompt_embeds-only 的请求在入队阶段就抛 AssertionError，
    # 最终以 EngineDeadError 的形式暴露给用户。
    # 现在该判断交由 pooling_runner 依据 pooling_params.requires_token_ids 完成。
    self.pooling_runner.add_request(
        req_id,
        req_index,
        new_req_data.pooling_params,
        new_req_data.prompt_token_ids, # 允许为 None
    )

```

评论区精华

本轮没有逐行评审意见 (review_comments_count = 0)，唯一实质动作是 njhill 先评论 `/ci run` 触发 Buildkite CI #88602 (commit `835c1975f9d2`)，随后以 APPROVED 无正文放行；PR 也因此以“修复 commit + 合并 main 的 merge commit”两个 commit 落地。claude[bot] 只留下一条流程性说明：这是来自 fork 的 PR，自动化审查被禁用，需要维护者评论 `@claude review` 才会执行一次性审查，本次未触发。因此“断言放在调用方还是使用侧”这一设计取舍没有被公开辩论，结论完全由作者在 PR 描述中给出并被维护者默认接受；此外，缺少自动化回归用例的疑虑也没有在讨论中被提出。

- 契约放宽的落点：断言留在调用方还是下沉到使用侧 (design): 方案被采纳；njhill 以 APPROVED 放行，无反对意见，现有 token 化请求与依赖 token id 的池化任务行为不变。
- CI 触发与合并流程 (other): CI 通过并由维护者批准，改动以修复 commit 加一次 main 合并提交落地。
- fork PR 上自动化审查被跳过 (other): 这段放宽入队契约的改动只经维护者人工批准，没有自动化深度审查；对 4 行改动而言影响有限。
- 缺少 prompt_embeds 场景的自动化回归用例 (testing): 未在本次补齐；建议后续在 vllm/v1/worker 或 pooling 集成测试中加入 MRV2 下 prompt_embeds 与普通 token 输入混合 batch 的用例，防止同类回归。

风险与影响

- 风险：1) 契约放宽带来的静默失败面：prompt_token_ids 变为可选后，校验完全依赖每个 pooler 的 requires_token_ids 声明。若某个池化任务实际需要 token id 却未声明该标志，错误会从入队期的断言（即时、可定位）推迟到 _get_pooling_metadata 或 pooler 内部，表现可能更隐蔽，例如构造出长度或内容不正确的输入。2) 无自动化回归测试：修复仅由手工 reproducer 与既有用例覆盖，仓库中没有新增覆盖“MRV2 下 prompt_embeds 与普通 token 输入混合 batch”的用例，也没有断言 prompt_embeds-only 请求不崩溃的测试文件变更，同类回归存在再次出现的可能。3) 混合批次路径需自查：self.prompt_token_ids 以 req_index 为键，批内可能同时存在需要与不需要 token id 的请求；当 any(params.requires_token_ids ...) 为真时，_get_pooling_metadata 会为整批构造 prompt_token_ids_cpu，此时批内缺失项如何对齐需要继续确认，属于本次改动后最值得复核的边界。4) 影响范围受限 opt-in 路径：MRV2 需显式 VLLM_USE_V2_MODEL_RUNNER=1，默认路径不受影响；但对该开关的已启用用户，原行为是崩溃级 (EngineDeadError)，因此这是修复而非新风险。
- 影响：对用户：开启 V2 model runner 的解码器池化用户 (Qwen3-Embedding-0.6B、BAAI/bge-multilingual-gemma2、intfloat/e5-mistral-7b-instruct) 可以用预计算 prompt_embeds 直接取嵌入，不必再伪造 token id；PR 的精度表显示 embeds 路径与 token id 路径最大绝对差为 0，说明计算结果等价。对系统：改动收敛在 worker 侧入队契约，model_runner.add_requests → pooling_runner.add_request 的职责边界更清晰——调用方只负责传递数据，是否必须存在由任务能力声明决定，后续新增池化任务时不必再改 model_runner。对团队：这是 MRV1 → MRV2 迁移中一个典型的 parity gap，提示迁移验证清单需要显式覆盖“非 token 输入” (prompt embeds、多模态、缓存等) 路径，而不能只跑文本 token 流程。影响程度整体为局部、低风险但修复价值明确。

- 风险标记：无新增回归测试，请求入队契约放宽，仅影响 MRV2 可选路径，崩溃级回归修复

关联脉络

- PR #56654 [MRV2] Revert explicit Triton JIT warmup migration: 同样直接修改 `vllm/v1/worker/gpu/model_runner.py`，属于 MRV2 代码路径的近期演进；两者都体现 MRV2 在追赶 MRV1 行为时的快速调整与回退节奏。
- PR #56573 [Pooling] Report actual input token usage for scoring APIs: 同属 pooling 请求链路（PoolingParams 与输入 token 相关契约）的近期改动；本 PR 放宽的 `requires_token_ids` 语义与该链路的输入契约同源，可对照理解池化任务的输入假设。