PR #47874 完整报告

vllm-project/vllm

[ROCm][CI][MoE] Fix double-transpose of fused w3 expert weights

合并时间：2026-07-09 07:59

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/47874

执行摘要

- 一句话：修复融合 MoE 专家权重 w3 双重转置崩溃
- 推荐动作：值得精读。这是一个典型的共享变量误用 bug，修复手法简洁高效，适合作为代码审查教学案例。建议开发者关注变量作用域和可变性对迭代逻辑的影响。

功能与动机

修复加载 Qwen/Qwen3-VL-30B-A3B-Instruct-FP8 等模型时，融合 MoE 专家权重 gate_up_proj 因双重转置导致 RuntimeError 的崩溃问题。PR body 明确指出 w3 分支看到的是 w1 已转置后的张量，导致第二次转置后 chunk 维度错误。

实现拆解

1. 共享变量加载问题：在 vllm/model_executor/layers/fused_moe/routed_experts.py 的 load_weights 方法中，融合张量 loaded_weight 在 expert_mapping 中会匹配两次（分别对应 w1 和 w3）。第一次匹配时，如果张量形状需要转置，loaded_weight 被直接赋值修改，导致第二次匹配时看到的是已转置的版本。
2. 局部变量修复：在 is_fused 分支开头引入局部变量 fused_weight = loaded_weight，后续转置和 chunk 操作均使用 fused_weight，确保 loaded_weight 保持原始状态。这样 w1 和 w3 的转置逻辑完全对称，互不干扰。
3. 非融合路径检查：w2（down_proj）分支只匹配一次，非融合路径（is_fused=False）使用 loaded_weight.unsqueeze(0)，两者均不受双重转置影响，因此无需修改。
4. 测试验证：PR 通过 pytest tests/v1/spec_decode/test_acceptance_length.py -k 'qwen3-30b-moe-vl-eagle3' 在 ROCm MI300 上通过端到端测试。

关键文件：

- vllm/model_executor/layers/fused_moe/routed_experts.py（模块 模型执行器；类别 source；类型 data-contract；符号 load_weights）：修复的核心文件，修改了 load_weights 方法中 fused 分支的变量作用域，避免 loaded_weight 被多次转置。

关键符号：load_weights

关键源码片段

vllm/model_executor/layers/fused_moe/routed_experts.py

修复的核心文件，修改了 load_weights 方法中 fused 分支的变量作用域，避免 loaded_weight 被多次转置。

```
def load_weights(
    self, weights: Iterable[tuple[str, torch.Tensor]]
) -> Iterable[str]:
    expert_mapping = self.get_expert_mapping(include_fused=True)
    unpadded_hidden = self.moe_config.hidden_dim_unpadded
    for expert_name, loaded_weight in weights:
        # ... 省略前面的匹配逻辑
        if is_fused:
            # w1 和 w3 共享同一个 fused 张量；使用局部变量 fused_weight
            # 防止下面的转置操作修改 loaded_weight，否则 w3 会看到
            # 已被 w1 转置过的版本，导致第二次转置和 chunk 维度错误
            fused_weight = loaded_weight
            if shard_id in {"w1", "w3"}:
                if fused_weight.shape[-1] != unpadded_hidden:
                    # [..., hidden, intermediate] -> [..., intermediate, hidden]
                    fused_weight = fused_weight.transpose(-1, -2)
                # 复用 expert_id 来解耦 w1 和 w3
                experts_shard = fused_weight.chunk(2, dim=1)[expert_id]
            else:
                # w2 分支，只需转置一次
                if fused_weight.shape[-2] != unpadded_hidden:
                    fused_weight = fused_weight.transpose(-1, -2)
                experts_shard = fused_weight
            # ... 后续统一加载逻辑
```

评论区精华

无 review 讨论。Approver AndreasKaratzas 确认测试通过，并请求 mgoin 审阅；zyongye 直接批准。

- 暂无高价值评论线程

风险与影响

- 风险：回归风险低：变更仅限于 fused MoE 权重的加载逻辑，引入了局部变量 fused_weight 替代直接修改 loaded_weight。该变量仅在 is_fused 分支中使用，不影响非融合路径和 w2 分支。PR 已通过目标模型端到端测试。唯一潜在风险是如果未来有代码依赖 loaded_weight 在迭代中被修改的行为，但当前代码无此依赖。
- 影响：正向影响：修复了 ROCm 平台上加载 Qwen3-VL 等 FP8 MoE 模型的崩溃问题，使这些模型在 ROCm 上可正常运行。影响范围限于融合 MoE 权重的加载路径，对 NVIDIA 和其他平台无负面影响。
- 风险标记：无实质性风险

关联脉络

- 暂无明显关联 PR