# PR #44912 完整报告

vllm-project/vllm

[Bugfix][ROCm] Fix FP8 per-tensor scale rank mismatch causing Inductor assertion failure

合并时间：2026-06-17 11:17

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/44912

## 执行摘要

- 一句话：修复 ROCm FP8 per-tensor scale 维度不匹配导致编译失败
- 推荐动作：推荐关注该 PR 的开发者阅读 apply_scaled_mm 中的注释和改动，以便理解 torch.compile 对 scale tensor rank 的严格约束。设计上选择 .view 而非 .reshape 体现了对性能的谨慎考量，值得借鉴。

## 功能与动机

修复 torch.compile 在 ROCm 上升级 aten._scaled_mm 时的断言失败。当 VLLM_ROCM_USE_AITER=1 且 VLLM_ROCM_USE_SKINNY_GEMM=0 时，requantize_with_max_scale 返回 0-D 的 weight scale，但 AITER 的 fake quant 返回 1-D activation scale，导致 torch.compile 在调整 kernel 输入时断言 len(scale_a.get_size()) == len(scale_b.get_size()) 失败，引发 InductorError: LoweringException: AssertionError。

## 实现拆解

1. 定位问题：分析发现 torch._scaled_mm 在 torch.compile 下要求 scale_a 和 scale_b 具有相同的张量秩（rank），而 PerTensorTorchFP8ScaledMMLinearKernel.apply_scaled_mm 中直接传递来自不同源的 scales，其维度可能不一致（0-D vs 1-D）。
2. 设计修复方案：在调用 torch._scaled_mm 之前，统一通过 .view(1) 将 0-D scales 提升为 1-D。.view() 不会复制数据，性能开销极小。最初尝试在 fp8.py 中的量化函数中修复，后经讨论决定收敛到 scaled_mm/pytorch.py 的 apply_scaled_mm 方法，使得 fix 更集中且不影响其他调用方。
3. 代码修改：在 apply_scaled_mm 的开头添加两行条件判断，仅当 As.dim() == 0 或 Bs.dim() == 0 时执行 .view(1)。该改动影响所有通过此 kernel 执行的 FP8 per-tensor 量化路径，但对非 0-D 场景无任何副作用。
4. 验证：通过修改后的代码在 MI355X 上加载 amd/Llama-3.1-8B-Instruct-FP8-KV 模型，成功完成编译和服务请求，未再出现 InductorError。
5. 测试与配置：当前改动无单独测试文件。但已在手动环境中验证正确性，且 CI 会覆盖相关配置（需待 CI 启用后自动验证）。

关键文件：

- vllm/model_executor/kernels/linear/scaled_mm/pytorch.py（模块 量化内核；类别 source；类型 data-contract；符号 apply_scaled_mm）：核心修改文件，在

PerTensorTorchFP8ScaledMMLinearKernel.apply_scaled_mm 中增加对 0-D scale 的维度归一化。

关键符号：apply_scaled_mm

## 关键源码片段

### vllm/model_executor/kernels/linear/scaled_mm/pytorch.py

核心修改文件，在 PerTensorTorchFP8ScaledMMLinearKernel.apply_scaled_mm 中增加对 0-D scale 的维度归一化。

```
# vllm/model_executor/kernels/linear/scaled_mm/pytorch.py
# PerTensorTorchFP8ScaledMMLinearKernel 类中的 apply_scaled_mm 方法
# 修复 torch.compile 下 0-D scale 导致 rank mismatch 的问题

    def apply_scaled_mm(
        self,
        *,
        A: torch.Tensor,
        B: torch.Tensor,
        out_dtype: torch.dtype,
        As: torch.Tensor,
        Bs: torch.Tensor,
        bias: torch.Tensor | None,
        output_shape: list,
    ) -> torch.Tensor:
        # torch._scaled_mm 在 torch.compile 下不支持 0-D scale 张量
        # 将 0-D 转换为 1-D (view 避免拷贝 )，确保 scale_a 和 scale_b 维度一致
        if As.dim() == 0:
            As = As.view(1)
        if Bs.dim() == 0:
            Bs = Bs.view(1)

        output = torch._scaled_mm(
            A, B, out_dtype=out_dtype, scale_a=As, scale_b=Bs, bias=bias
        )
        # 兼容 torch < 2.5 返回 tuple 的情况
        if type(output) is tuple and len(output) == 2:
            output = output[0]

        num_tokens = _get_num_tokens(output_shape)
        return torch.narrow(output, 0, 0, num_tokens).view(*output_shape)
```

## 评论区精华

在 Issue 评论中，AndreasKaratzas 提出使用 .view 代替 .reshape 以避免拷贝，divakar-amd 确认可行并解释 view 会隐式调用不会复制。最终采用 .view(1)。此外，fix 的位置最初在 fp8.py，后经 tjtanaa 和 divakar-amd 建议移至 scaled_mm/pytorch.py，使修复更集中。

- 使用 view 替代 reshape 避免拷贝 (performance): 使用 .view(1) 而不是 .reshape(1)，因为 view 不产生数据拷贝。
- 修复位置从 fp8.py 迁移至 scaled_mm/pytorch.py (design): 统一在 apply_scaled_mm 中做维度归一化，不影响其他路径。

## 风险与影响

- 风险：该修复仅向 apply_scaled_mm 方法添加了两行维度检查，将 0-D tensor 提升为 1-D。风险极低：
  - 对非 0-D scales 完全无影响。
  - view 仅改变元数据，不产生数据拷贝，性能无退化。
  - 改动局限于 FP8 per-tensor 量化路径，不影响其他精度或后端。
  - 潜在风险：如果未来有新的 scale tensor 生成路径返回 0-D 但语义上不应用 1-D 表示（可能性极小），此 fix 可能掩盖问题。但当前所有 0-D 场景均正确表示 per-tensor scale，1-D 在语义上等价。
  - 缺少自动化测试覆盖该特定 ROCm 编译路径，回归检测依赖手动测试和 CI。
  - 影响：
  - 用户影响：修复了 ROCm 上特定配置（AITER + 禁用 skinny GEMM + torch.compile）的启动崩溃，使这些用户可以正常使用 FP8 per-tensor 量化模型。对其他配置无影响。
  - 系统影响：无。改动极其局部，不影响系统整体架构。
  - 团队影响：低。提交者与 reviewer 已达成一致，合并过程顺利。
  - 风险标记：torch.compile 兼容性，0-D 张量处理

## 关联脉络

- PR #41293 [Bugfix][ROCm] Fix FP8 scale rank mismatch under torch.compile: 同一问题的先前尝试 PR，被关闭并被本 PR 替代。本 PR 在其基础上调整为在 scaled_mm/pytorch.py 中修复。