PR #45935 完整报告

vllm-project/vllm

[Model]Fix MiniMaxM2ForCausalLM perf regression

合并时间：2026-06-22 00:28

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/45935

执行摘要

- 一句话：Triton 融合替代 torch.compile 修复 MiniMax 性能回归
- 推荐动作：值得精读，尤其关注如何用 Triton 手动替代 torch.compile 进行算子融合的方案，可作为类似性能优化案例的参考。

功能与动机

PR body 指出：The root cause is that torch.compile can't fuse these torch glue ops, which leads to the M25 perf regression. This PR fuses these torch ops manually with Triton.

实现拆解

实现分为四步：

1. 新增两个 Triton JIT kernel（rms_norm_tp.py）：_minimax_qk_var_kernel 计算每个 token 中 q 和 k 分段的均方和，_minimax_rms_apply_kernel 使用 all-reduce 后的方差对 q 和 k 进行归一化并与权重相乘。
2. 新增调度函数_minimax_qk_norm_tp_fallback：先调用 var kernel，然后执行 all-reduce 方差，再调用 apply kernel。同时新增 _minimax_qk_norm_tp_eager 作为纯 PyTorch 实现，用于精度参考和回退。
3. 删除 @torch.compile 装饰器：原 _minimax_qk_norm_fallback 函数体替换为调用 _minimax_qk_norm_tp_fallback（当 fused kernel 不可用或超限时），使所有非 fused 路径都经由 Triton kernel。
4. 更新测试（test_minimax_reduce_rms.py）：参考路径改为调用 _minimax_qk_norm_tp_eager 保持一致性；新增 test_minimax_qk_norm_triton_fallback 单 GPU 测试，通过 monkeypatch all-reduce 验证 Triton 计算的正确性。

关键文件：

- vllm/model_executor/layers/minimax_rms_norm/rms_norm_tp.py（模块 归一化层；类别 source；类型 core-logic；符号 _minimax_qk_norm_fallback, _minimax_qk_var_kernel, _minimax_rms_apply_kernel, _minimax_qk_norm_tp_eager）：核心变更：将 QK RMS-norm 的 fallback 路径从 torch.compile 替换为手写 Triton kernel，消除融合失败导致的性能损失。

- tests/kernels/core/test_minimax_reduce_rms.py（模块 单元测试；类别 test；类型 test-coverage；符号 test_minimax_qk_norm_triton_fallback）：修改多 GPU 测试参考路径，新增单 GPU Triton fallback 测试，确保 kernel 的数值正确性。

关键符号：_minimax_qk_norm_fallback, _minimax_qk_var_kernel, _minimax_rms_apply_kernel, _minimax_qk_norm_tp_eager, _minimax_qk_norm_tp_fallback, test_minimax_qk_norm_triton_fallback

关键源码片段

vllm/model_executor/layers/minimax_rms_norm/rms_norm_tp.py

核心变更：将 QK RMS-norm 的 fallback 路径从 torch.compile 替换为手写 Triton kernel，消除融合失败导致的性能损失。

```
# SPDX-License-Identifier: Apache-2.0
# SPDX-FileCopyrightText: Copyright contributors to the vLLM project

from vllm.triton_utils import HAS_TRITON, tl, triton


@triton.jit
def _minimax_qk_var_kernel(
    qkv_ptr, # [num_tokens, hidden], 16-bit 激活值
    var_ptr, # [num_tokens, 2], fp32 输出方差
    row_stride, # qkv 中 token 间的元素跨度
    q_size: tl.constexpr,
    kv_size: tl.constexpr,
    BLOCK: tl.constexpr,
):
    '''TP 前处理：计算每个 token 中 q 和 k 分段的均方和（方差分量）。

    直接在原 16-bit qkv 上读取并以 fp32 累加，避免产生 fp32 的 q/k 拷贝。
    var[:, 0] 对应 q 方差，var[:, 1] 对应 k 方差；
    均为本地分片的均值，后续进行 all-reduce。
    '''
    token = tl.program_id(0)
    base = qkv_ptr + token * row_stride

    q_acc = 0.0
    for off in range(0, q_size, BLOCK):
        idx = off + tl.arange(0, BLOCK)
        mask = idx < q_size
        x = tl.load(base + idx, mask=mask, other=0.0).to(tl.float32)
        q_acc += tl.sum(x * x, axis=0)

    k_acc = 0.0
    for off in range(0, kv_size, BLOCK):
        idx = off + tl.arange(0, BLOCK)
        mask = idx < kv_size
```

```
        x = tl.load(base + q_size + idx, mask=mask, other=0.0).to(tl.float32)
        k_acc += tl.sum(x * x, axis=0)

    tl.store(var_ptr + token * 2 + 0, q_acc / q_size)
    tl.store(var_ptr + token * 2 + 1, k_acc / kv_size)
```

评论区精华

PR 由 ZJY0516 批准，未产生审核评论。

- 暂无高价值评论线程

风险与影响

- 风险：Triton kernel 仅在 CUDA 且安装 Triton 时可用，通过 HAS_TRITON 守卫；非 CUDA 平台将自动回退到 *eager* 路径，不影响正确性但可能无性能收益。单 GPU 测试通过 monkeypatch 覆盖了多 rank 的 scaling 逻辑，但缺少多 GPU 端到端集成测试。数值精度已通过官方评测基准（GPQA Diamond 0.833 vs 0.839，AIME 2025 0.858 vs 0.853）验证基本一致，但在非标准场景下需额外验证。
- 影响：仅影响 MiniMaxM2.5（及其他使用相同 QK RMS-norm 的 MiniMax 变体）的推理性能，预期在批量大小适中时显著提升 token 生成速度。其他模型不受影响。对于使用 MiniMaxM2.5 的生产部署，建议在包含 CUDA 和 Triton 的环境中运行以获得最佳性能。
- 风险标记：依赖 Triton，多 GPU 测试覆盖不完全，数值精度依赖 benchmark 验证

关联脉络

- 暂无明显关联 PR