PR #44893 完整报告

vllm-project/vllm

[ROCm][gpt-oss] Pass GateMode.INTERLEAVE for MXFP4 W4A16 fused MoE

合并时间：2026-06-12 14:02

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/44893

执行摘要

- 一句话：修复 MXFP4 W4A16 在 AITER 新版本上的准确率归零
- 推荐动作：建议阅读者关注本次 PR 中 gate_mode 与权重 shuffle 一致性的设计，以及通过 inspect.signature 实现运行时兼容性探测的工程技巧。该修复是 AITER 版本演进中保持兼容性的好例子。

功能与动机

AITER 的 PR#3123 为 fused MoE 增加了 gate_mode 参数，但 vllm 调用 fused_moe 时未传递。对于 MXFP4 W4A16，权重在 shuffle 时已 interleave，因此需告知 kernel 采用 INTERLEAVE 模式。缺失该参数导致 dispatcher 进入错误的 kernel 路径，产生全零输出（gpt-oss-120b acc=0.0）或 CK2stages JIT 崩溃（gpt-oss-20b）。Issue#3586 详细报告了该问题。

实现拆解

1. 接口扩展：在 vllm/_aiter_ops.py 中，为 _rocm_aiter_fused_moe_impl 和 fused_moe 方法添加 gate_mode: str = "" 参数。在内部，当 gate_mode 非空且探测到 fused_moe_supports_gate_mode() 返回 True 时，通过 **extra_kwargs 将 gate_mode 传递给底层 aiter.fused_moe。
2. 兼容性探测：新增 fused_moe_supports_gate_mode 类方法，利用 inspect.signature 检查当前安装的 aiter.fused_moe 函数签名是否包含 gate_mode 参数，并缓存结果，以兼容未引入该参数的旧版 AITER。
3. 调用点注解：在 vllm/model_executor/layers/fused_moe/experts/rocm_aiter_moe.py 的 rocm_aiter_fused_experts 函数中，当 quant_config.use_mxfp4_w4a16 为 True 时，尝试从 aiter.ops.flydsl.moe_common 导入 GateMode 并设置 gate_mode = GateMode.INTERLEAVE.value，若导入失败（旧版 AITER）则静默忽略。
4. 传递 gate_mode：在调用 rocm_aiter_ops.fused_moe 时显式传入 gate_mode=gate_mode，以确保 kernel 按 INTERLEAVE 模式处理 gate 和 up 权重，恢复正确的 SwiGLU 计算。
5. 验证：PR body 报告在 MI355X (gfx950) 上，gpt-oss-120b W4A16 的 gsm8k 准确率从 0.0 恢复至 0.9+；gpt-oss-20b 的 CK2stages JIT 崩溃已解决，且在多个 AITER 版本上均验证通过。无新增自动化测试，仅手工验证。

关键文件：

- vllm/_aiter_ops.py（模块 AITER 桥接层；类别 source；类型 core-logic；符号 fused_moe_supports_gate_mode, _rocm_aiter_fused_moe_impl, fused_moe）：核心 ops 层：新增 gate_mode 参数和向后兼容探测逻辑。
- vllm/model_executor/layers/fused_moe/experts/rocm_aiter_moe.py（模块 MoE 专家层；类别 source；类型 data-contract）：MoE 专家层：在 MXFP4 W4A16 路径下设置 GateMode.INTERLEAVE。

关键符号：fused_moe_supports_gate_mode, _rocm_aiter_fused_moe_impl, fused_moe, rocm_aiter_fused_experts

关键源码片段

vllm/_aiter_ops.py

核心 ops 层：新增 gate_mode 参数和向后兼容探测逻辑。

```
# vllm/_aiter_ops.py (head 版本关键片段 )

def _rocm_aiter_fused_moe_impl(
    hidden_states: torch.Tensor,
    w1: torch.Tensor,
    w2: torch.Tensor,
    topk_weight: torch.Tensor,
    topk_ids: torch.Tensor,
    expert_mask: torch.Tensor | None = None,
    activation_method: int = 0,
    quant_method: int = 0,
    doweight_stage1: bool = False,
    w1_scale: torch.Tensor | None = None,
    w2_scale: torch.Tensor | None = None,
    a1_scale: torch.Tensor | None = None,
    a2_scale: torch.Tensor | None = None,
    num_local_tokens: torch.Tensor | None = None,
    output_dtype: torch.dtype | None = None,
    hidden_pad: int = 0,
    intermediate_pad: int = 0,
    gate_mode: str = "", # 新增：门模式，由调用方传入
    bias1: torch.Tensor | None = None,
    bias2: torch.Tensor | None = None,
    moe_sorting_dispatch_policy: int = 0,
) -> torch.Tensor:
    from aiter import ActivationType, QuantType
    from aiter.fused_moe import fused_moe

    activation = ActivationType(activation_method)
    quant_type = QuantType(quant_method)

    # 仅在 gate_mode 非空且当前 AITER 版本支持 gate_mode 时传递
```

```
    extra_kwargs: dict = {}
    if gate_mode and rocm_aiter_ops.fused_moe_supports_gate_mode():
        extra_kwargs["gate_mode"] = gate_mode

    return fused_moe(
        hidden_states,
        w1,
        w2,
        topk_weight,
        topk_ids,
        expert_mask,
        activation,
        quant_type,
        doweight_stage1,
        w1_scale,
        w2_scale,
        a1_scale,
        a2_scale,
        num_local_tokens=num_local_tokens,
        dtype=output_dtype,
        hidden_pad=hidden_pad,
        intermediate_pad=intermediate_pad,
        bias1=bias1,
        bias2=bias2,
        moe_sorting_dispatch_policy=moe_sorting_dispatch_policy,
        **extra_kwargs, # 条件展开 gate_mode
    )


class _AiterOps:
    # ... 其他方法 ...

    @classmethod
    @if_aiter_supported
    @functools.cache
    def fused_moe_supports_gate_mode(cls) -> bool:
        """
        探查已安装的 aiter.fused_moe 是否接受 gate_mode 参数。
        从 https://github.com/ROCm/aiter/pull/3123 (>=0.1.14) 开始支持。
        旧版本必须省略此参数以避免 TypeError。
        """
        import inspect
        from aiter.fused_moe import fused_moe

        return "gate_mode" in inspect.signature(fused_moe).parameters
```

vllm/model_executor/layers/fused_moe/experts/rocm_aiter_moe.py

MoE 专家层：在 MXFP4 W4A16 路径下设置 GateMode.INTERLEAVE。

```
# vllm/model_executor/layers/fused_moe/experts/rocm_aiter_moe.py (head 版本关键片段 )

# 在 rocm_aiter_fused_experts 计算 padding 后的部分:

# AITER 从 PR#3123 开始将 stage1 GEMM 分为 interleaved 和 separated 两条路径。
# 对于 gpt-oss 即 use_mxfp4_w4a16=True，权重由 `shuffle_weight_a16w4` 以
# is_guinterleave=True 方式 shuffle，因此必须传递 GateMode.INTERLEAVE。
gate_mode = ""
if quant_config.use_mxfp4_w4a16:
    try:
        from aiter.ops.flydsl.moe_common import GateMode
        gate_mode = GateMode.INTERLEAVE.value
    except ImportError:
        # 旧版 AITER 无 GateMode，静默跳过
        pass

return rocm_aiter_ops.fused_moe(
    hidden_states,
    w1,
    w2,
    topk_weights,
    topk_ids,
    expert_mask=expert_mask,
    quant_method=quant_method,
    activation_method=activation_method,
    w1_scale=quant_config.w1_scale,
    w2_scale=quant_config.w2_scale,
    a1_scale=quant_config.a1_scale if a1q_scale is None else a1q_scale,
    a2_scale=quant_config.a2_scale,
    doweight_stage1=apply_router_weight_on_input,
    num_local_tokens=num_local_tokens,
    output_dtype=output_dtype,
    hidden_pad=hidden_pad,
    intermediate_pad=intermediate_pad,
    gate_mode=gate_mode, # 传递门模式
    bias1=quant_config.w1_bias if quant_config.use_mxfp4_w4a16 else None,
    bias2=quant_config.w2_bias if quant_config.use_mxfp4_w4a16 else None,
    moe_sorting_dispatch_policy=moe_sorting_dispatch_policy,
)
```

评论区精华

PR 的 review 讨论较少，主要由作者在 PR body 和关联 issue 中详细分析了根因并给出了验证矩阵。Reviewer AndreasKaratzas 已批准。开发者 akii96 在评论中确认该修复解决了新版本 AITER 的准确率问题，期望尽快合入。PR 通过 mergify pre-commit 检查。没有显著的设计争议。

- 测试验证确认 (testing): 确认修复有效，期待合并。

风险与影响

- 风险：兼容性风险：通过运行时探测 aiter.fused_moe 签名，旧版 AITER 不受影响。 回归风险：仅影响 use_mxfp4_w4a16=True 的路径，其他 MoE 路径（W4A8、W8A8 等）不变，且已验证多种配置（TP=1/8、enforce-eager）。 测试覆盖：缺少单元测试，依赖手工验证，下次 AITER 版本升级可能引入新问题。 性能风险：仅增加一次 inspect 调用并缓存，无额外运行时开销。
- 影响：用户：使用 ROCm 平台、GPT-OSS MXFP4 W4A16 模型的用户将能从准确率归零恢复到正常（gsm8k > 0.9）。其他用户无影响。 系统：无系统级影响。 团队：低风险，维护负担低，代码变更聚焦。
- 风险标记：缺少测试覆盖，兼容性依赖，特定硬件路径

关联脉络

- 暂无明显关联 PR