PR #43781 完整报告

vllm-project/vllm

[Bugfix][ROCm] Fix Accuracy Drop in Sparse Indexer on gfx950

合并时间：2026-05-28 18:37

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/43781

执行摘要

- 一句话：修复 ROCm gfx950 上稀疏索引器精度问题
- 推荐动作：此 PR 值得关注，特别是 ROCm 开发和模型精度维护团队。代码改动简洁，但涉及 ROCm 稀疏索引器的核心逻辑，建议确保所有相关 kernel 配置匹配，并补充自动化测试。

功能与动机

ROCm 的稀疏索引器路径已支持 DeepSeek-V4，但当前配置导致旧模型（如 DeepSeek-V3.2 和 GLM-5）在 gfx950 上精度下降。PR body 明确指出有两个问题：1）索引器 cache layout 硬编码为 SHUFFLE，当 block_size=1 时导致错误缓存读取；2）ROCm 快速路径依赖 RoPE 实现原地修改输入张量，但 TorchInductor 使用的 PyTorch 原生 RoPE 并非原地操作。

实现拆解

1. 在 Indexer 类中新增 is_inplace_rope 开关（vllm/model_executor/models/deepseek_v2.py）：在 __init__ 方法中添加 is_inplace_rope: bool = False 参数，并在 forward 方法中将 if current_platform.is_rocm(): 条件改为 if current_platform.is_rocm() and self.is_inplace_rope:。同时传递注释说明快速路径依赖原地修改，而 PyTorch 原生 RoPE 返回新张量。
2. 在创建 Indexer 实例时传递 is_inplace_rope（同上文件第 1009 行附近）：调用 self.indexer_rope_emb.enabled() 判断是否使用 inplace 自定义算子，并将结果传入 Indexer。
3. 修复 cache layout 逻辑（vllm/v1/attention/ops/rocm_aiter_mla_sparse.py）：在 indexer_k_quant_and_cache_triton 和 cp_gather_indexer_k_quant_cache_triton 函数中，将硬编码的 "SHUFFLE" 替换为根据 block_size 动态选择的变量 layout（当 block_size == 1 时使用 "NORMAL"，否则使用 "SHUFFLE"）。同时修改 rocm_fp8_paged_mqa_logits 函数中的 Preshuffle 参数条件，从 block_size == 64 改为 block_size > 1。

关键文件：

- vllm/model_executor/models/deepseek_v2.py（模块 模型层；类别 source；类型 core-logic）：核心变更文件：新增 is_inplace_rope 参数并在 forward 中根据其值决定 ROCm 快速路径；修改 Indexer 初始化调用处传递该参数。

- vllm/v1/attention/ops/rocm_aiter_mla_sparse.py（模块 注意力层；类别 infra；类型 infrastructure）：次要变更：修复 cache layout 硬编码问题，根据 block_size 动态选择 NORMAL 或 SHUFFLE；修正 Preshuffle 条件。

关键符号：Indexer.init, Indexer.forward, indexer_k_quant_and_cache_triton, cp_gather_indexer_k_quant_cache_triton, rocm_fp8_paged_mqa_logits

关键源码片段

vllm/model_executor/models/deepseek_v2.py

核心变更文件：新增 is_inplace_rope 参数并在 forward 中根据其值决定 ROCm 快速路径；修改 Indexer 初始化调用处传递该参数。

```
# vllm/model_executor/models/deepseek_v2.py

class Indexer(nn.Module):
    def __init__(
        self,
        vllm_config: VllmConfig,
        config: DeepseekV2Config | DeepseekV3Config,
        hidden_size: int,
        q_lora_rank: int,
        quant_config: QuantizationConfig | None,
        cache_config: CacheConfig | None,
        topk_indices_buffer: torch.Tensor | None,
        prefix: str = "",
        is_inplace_rope: bool = False, # 新增：指示 RoPE 实现是否原地修改张量
    ):
        # ... 原有初始化代码 ...
        self.is_inplace_rope = is_inplace_rope # 存储该开关

    def forward(
        self, hidden_states: torch.Tensor, qr: torch.Tensor, positions, rotary_emb
    ) -> torch.Tensor:
        q, _ = self.wq_b(qr)
        q = q.view(-1, self.n_head, self.head_dim)

        # 原条件：if current_platform.is_rocm():
        # 修改为只有当 is_inplace_rope 为 True 时才走快速路径
        if current_platform.is_rocm() and self.is_inplace_rope:
            # 快速路径：依赖 rotary_emb 原地修改 q 和 k，避免 split/cat/copy
            kw, _ = self.wk_weights_proj(hidden_states)
            k = kw[:, : self.head_dim]
            weights = kw[:, self.head_dim :]
            k = self.k_norm(k)
            rotary_emb(
                positions, q[..., : self.rope_dim], k[..., : self.rope_dim].unsqueeze(1)
            )
        else:
```

```
        # 通用路径：先 split q_pe/q_nope，再应用 RoPE（返回新张量）
        q_pe, q_nope = torch.split(
            q, [self.rope_dim, self.head_dim - self.rope_dim], dim=-1
        )
        kw, _ = self.wk_weights_proj(hidden_states)
        k = kw[:, : self.head_dim]
        weights = kw[:, self.head_dim :]
        k = self.k_norm(k)
        k_pe, k_nope = torch.split(
            k, [self.rope_dim, self.head_dim - self.rope_dim], dim=-1
        )
        q_pe, k_pe = rotary_emb(positions, q_pe, k_pe.unsqueeze(1))
        # ... 后续合并逻辑 ...
```

vllm/v1/attention/ops/rocm_aiter_mla_sparse.py

次要变更：修复 cache layout 硬编码问题，根据 block_size 动态选择 NORMAL 或 SHUFFLE；修正 Preshuffle 条件。

```
# vllm/v1/attention/ops/rocm_aiter_mla_sparse.py

def indexer_k_quant_and_cache_triton(
    # ... 参数 ...
):
    k = k.view(torch.float8_e4m3fnuz) # 或者其他 fp8 dtype
    kv_cache_value = kv_cache[:, : block_size * head_dim].view(fp8_dtype)
    kv_cache_scale = kv_cache[:, block_size * head_dim :].view(torch.float32)
    head_tile_size = head_tile_size // kv_cache.element_size()
    # 新增：根据 block_size 决定 layout
    layout = "NORMAL" if block_size == 1 else "SHUFFLE"
    grid = (num_tokens,)
    _indexer_k_quant_and_cache_kernel[grid](
        k,
        # ... 其他参数 ...
        "SHUFFLE", # 原硬编码，现改为 layout
        layout, # 替换为动态变量
        block_tile_size,
        head_tile_size,
        IS_FNUZ=current_platform.fp8_dtype() == torch.float8_e4m3fnuz,
        # ...
    )
```

评论区精华

未发现讨论，仅有一条 LGTM 的审批评论。

- 暂无高价值评论线程

风险与影响

- 风险：此 PR 的更改范围小且逻辑清晰，主要风险在于：1）新增的 is_inplace_rope 条件判断可能遗漏某些自定义算子场景，导致仍然出现精度问题；2）layout 逻辑的调整可能影响其他 block_size 组合（如 block_size=128 等），需确保所有相关 kernel 的 layout 选择正确；3）无对应测试文件，回归验证依赖于人工跑 lm_eval，可能遗漏边界情况。
- 影响：影响范围：仅影响 ROCm 平台（gfx950）、DeepSeek-V3.2 和 GLM-5 等使用稀疏索引器的模型。精度提升显著（GSM8K 从 0.0129 提升至 0.9507），对使用 MLAAttention 和 SparseAttnIndexer 的模型有正面影响；对于使用 inplace RoPE 自定义算子的场景（如 DeepSeek-V4），行为不变。测试覆盖尚需加强。
- 风险标记：缺少测试覆盖，ROCm 特定路径

关联脉络

- PR #43679 [ROCm][DSV4] Enable Tilelang MHC replacing torch/triton mhc: 同为 ROCm 和 DeepSeek-V4 相关，涉及 ROCm 上的 kernel 优化和精度维护。
- PR #43829 [DSV4] Remove AMD/XPU path in deepseek_v4/nvidia: 涉及 DeepSeek-V4 的 AMD 路径清理，可能与稀疏索引器有间接交互。