PR #28787 完整报告

sgl-project/sglang

[AMD] Fix RMSNorm batch-invariance on ROCm under deterministic inference

合并时间：2026-07-05 14:49

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28787

执行摘要

- 一句话：修复 ROCm RMSNorm 批次不变性
- 推荐动作：值得合并。这是一个针对 ROCm 平台的精确 bugfix，修复了确定性推理模式下 RMSNorm 的批次不变性问题，同时通过使用 Triton 核函数避免了性能回退。评审过程健康，反馈被及时采纳。无需精读。

功能与动机

修复 ROCm 上 --enable-deterministic-inference 下 RMSNorm 的批次不变性问题。vLLM/aiter 的 fused 核函数在归一化约简时依赖 batch 形状，导致相同行在不同 batch 大小下输出略有差异，表现为 logprob 抖动。forward_cuda 已通过回退原生路径解决，而 forward_hip 缺少此保护。

实现拆解

1. 在 forward_hip 中添加批次不变性保护（python/sglang/srt/layers/layernorm.py）：当 is_batch_invariant_mode_enabled() 返回 True 时，检查是否可以使用高性能的 rms_norm_batch_invariant 核函数（条件：residual 为 None、未启用 cast_x_before_out_mul、且未使用 FSDP RL 策略）。满足条件则调用 Triton 实现的批次不变核函数；否则回退到 forward_native。
2. 在 forward_aiter 中添加批次不变性保护（同一个文件）：类似地，在 aiter 路径上也增加批次不变性检查，但额外检查是否使用了 fused pad kernel（self._fused_pad_kernel 且 self.x_pad_to_multiple > 0），此时直接回退原生路径，避免无必要的复杂判断。
3. 代码审查反馈与优化：初始版本中 forward_hip 直接无条件回退原生路径，性能较差。根据代码审查建议，改为镜像 forward_cuda 的优化路径，优先使用 Triton 批次不变核函数。同时 aiter 路径也在同一审查意见下得到补充保护。

关键文件：

- python/sglang/srt/layers/layernorm.py（模块 算子层；类别 source；类型 core-logic；符号 forward_hip, forward_aiter）：包含所有修改：在 forward_hip 和 forward_aiter 中添加批次不变性保护逻辑，是该 PR 的核心变更文件。

关键符号：forward_hip, forward_aiter

关键源码片段

python/sglang/srt/layers/layernorm.py

包含所有修改：在 forward_hip 和 forward_aiter 中添加批次不变性保护逻辑，是该 PR 的核心变更文件。

```
# python/sglang/srt/layers/layernorm.py 中的 forward_hip 方法（部分） defforward_hip( self, x: torch.Tensor, residual: Optional[torch.Tensor] = None, post_residual_addition: Optional[torch.Tensor] = None, ) -> Union[torch.Tensor, Tuple[torch.Tensor, torch.Tensor]]: # 首先回退到 native 实现（如果 vllm 不可用） if not _has_vllm_rms_norm: return self.forward_native(x, residual, post_residual_addition) # 批次不变性检查：如果启用确定性推理且符合条件，使用 Triton 批次不变核或回退 native if is_batch_invariant_mode_enabled(): if ( residual is not None or self.cast_x_before_out_mul or get_global_server_args().rl_on_policy_target == "fsdp" ): # 这些情况下无法安全使用 Triton 批次不变核，直接回退 native return self.forward_native(x, residual, post_residual_addition) # 使用 Triton 实现的批次不变 RMSNorm 核函数，性能更优 return rms_norm_batch_invariant( x, self.weight.data, self.variance_epsilon, ) # 以下为原有逻辑，仅在非批次不变模式下执行 if not x.is_contiguous(): x = x.contiguous() if residual is not None: out = torch.empty_like(x) residual_out = torch.empty_like(x) if post_residual_addition is not None: residual = residual + post_residual_addition fused_add_rms_norm( out, x, residual_out, residual, self.weight.data, self.variance_epsilon ) return out, residual_out out = torch.empty_like(x) rms_norm(out, x, self.weight.data, self.variance_epsilon) return out # forward_aiter 方法中的批次不变性保护（部分） defforward_aiter(self, ...): # ... 先前的连续性处理和 reshape 逻辑 ... if is_batch_invariant_mode_enabled(): if ( residual is not None or self.cast_x_before_out_mul or get_global_server_args().rl_on_policy_target == "fsdp" or (self._fused_pad_kernel is not None and self.x_pad_to_multiple > 0) ): return self.forward_native(x, residual, post_residual_addition) out = rms_norm_batch_invariant( x, self.weight.data, self.variance_epsilon, ) if needs_reshape: out = out.reshape(original_shape) return out # ... 后续原有 fused 和 aiter 逻辑 ...
```

评论区精华

代码审查（gemini-code-assist[bot]）：建议不要在批次不变模式下无条件回退到慢速 forward_native，而应镜像 forward_cuda 的优化逻辑，使用 Triton 的 rms_norm_batch_invariant 核函数。开发者采纳此建议，在后续 commit 中改为优先使用高性能 Triton 核函数。

Reviewer 1am9trash：指出现有 PR 只保护了 forward_hip，提醒也应保护 forward_aiter 路径。开发者随后在 commit 26a3a5ec 中增加了对 aiter 路径的保护。

- forward_hip 应使用 Triton 批次不变核而非无条件回退 native (performance): 开发者采纳建议，在后续 commit 中改为优先使用 Triton 核函数，仅在特定条件不满足时回退 native。

- forward_aiter 也应添加批次不变性保护 (correctness): 开发者在后续 commit 中为 forward_aiter 添加了相同的批次不变性检查。

风险与影响

- 风险:
  - 回归风险：低。本次变更仅在 --enable-deterministic-inference 模式下生效，非该模式下行为完全不变。
  - 性能风险：低。在批次不变模式下，优先使用 Triton 的 rms_norm_batch_invariant 核函数，性能优于无条件回退原生路径。
  - 兼容性风险：低。仅影响 ROCm 平台，且仅在确定性推理模式下改变行为。
  - 缺少测试覆盖：本次 PR 未添加新的单元测试，仅依赖现有 CI（AMD nightly）覆盖该代码路径。
- 影响:
  - 用户：使用 ROCm 且开启 --enable-deterministic-inference 的用户将获得正确的批次不变性，消除不同 batch size 下的 logprob 抖动。
  - 系统：无全局影响，仅修改了 layernorm.py 一个文件。
  - 团队：需确保 AMD CI 中此变更路径被充分测试。
  - 风险标记：缺少测试覆盖

关联脉络

- 暂无明显关联 PR