

PR #52208 完整报告

vllm-project/vllm

[ROCm][CI] add Aiter ops tests

合并时间: 2026-08-18 07:33

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/52208>

执行摘要

- 一句话: 新增 AITER 核心算子测试, MI355 镜像开启算子门禁
- 推荐动作: 值得精读, 尤其建议关注两点: 一是 `_assert_close_budget / _assert_rel_error_quality` 的预算式数值断言设计, 比裸 `allclose` 更适合 kernel 级测试的统计抖动, 可推广到其他算子测试; 二是 Rohan138 关于 vLLM C++ 与 Triton RoPE 精度行为分歧的评论, 揭示了同一功能在不同后端实现间缺乏统一数值语义的系统性问题, 对后续 kernel 选型与测试编写都有参考价值。

功能与动机

PR body 明确指出目标是 'Add ROCm AITER ops tests and enable Kernels Core Operation Test for the MI355 mirror', 并说明这些测试 'directly validate AITER kernel correctness against PyTorch/reference implementations', 其价值在于 'provide clear failure signals when AITER library updates introduce regressions, independent of vLLM's fusion/compilation passes'。即在现有端到端测试无法定位 kernel 层回归的背景下, 建立独立于融合 / 编译链路的算子级健康检查。

实现拆解

实现分为三层: 源码封装、测试套件、CI 门禁, 彼此联动。

1. 在 `rocm_aiter_ops` 中新增两个 AITER 封装入口

- 文件 `vllm/_aiter_ops.py` 的 `rocm_aiter_ops` 类新增静态方法 `rms_norm` 与 `rms_norm2d_with_add`。
- `rms_norm` 直接转发到 `aiter.rms_norm`; `rms_norm2d_with_add` 先通过 `torch.empty_like` 分配输出缓冲, 再调用 `aiter.rmsnorm2d_fwd_with_add` 就地写入, 返回 `(normalized_output, residual_sum)` 二元组。
- 采用延迟 `import aiter`, 使模块在未安装 AITER 的环境下仍可正常导入; 两个方法仅作为测试入口, 暂不进入生产调用路径。

2. 新增 ROCm AITER 核心算子测试套件

- 新文件 `tests/kernels/core/test_rocm_aiter_ops.py`, 共 625 行, 覆盖四类场景: `rms_norm` 系 (含 `rms_norm2d_with_add` 残差融合)、`triton_rotary_embedding` (NeoX 风格, `stict xfail`)、`act_mul_and_fp8_group_quant` (SiGLU + FP8 分组量化)、

fused RMSNorm + 量化 vs 顺序组合的端到端链路。

- 提供两套可复用的数值断言工具：_assert_close_budget 采用 pass_rate 预算 + max/mean/p99/p999/worst_ratio 统计量，允许少量元素超出 atol 并输出完整诊断信息；_assert_rel_error_quality 针对相对误差设置 mean_limit、preferred_rel、max_rel 与 max_fail_rate 四重约束。
- 通过 require_aiter、require_fp8 对不支持 AITER 或 FP8 的硬件（如 mi250）显式 skip，fixture setup_cuda_device 统一设置默认设备与随机种子。

3. 在 CI 中启用 Kernels Core Operation Test

- .buildkite/test-amd.yaml 在 mi355 镜像分组中新增 label 为 Kernels Core Operation Test %N 的 job，parallelism 设为 3，按 BUILDKITE_PARALLEL_JOB 分片执行 pytest tests/kernels/core（排除 test_minimax_reduce_rms.py）及 test_concat_mla_q.py、test_top_k_per_row.py。
- source_file_dependencies 覆盖 csrc/ 全量、tests/kernels/core、rotary_embedding 实现与 vllm/_aiter_ops.py，确保 kernel 源码或封装变更时自动触发该门禁。

测试、源码、CI 三者形成闭环：源码封装为测试提供入口，测试结果作为 AITER 更新与 kernel 改动的回归信号，CI 配置将信号接入 MI355 镜像的门禁队列。

关键文件：

- tests/kernels/core/test_rocm_aiter_ops.py（模块 算子测试；类别 test；类型 test-coverage；符号 setup_cuda_device, require_aiter, require_fp8, _format_observed_rate）：新增 625 行 AITER 算子测试主体，覆盖 RMSNorm、残差融合、FP8 分组量化、RoPE 与端到端链路，并内置预算式数值断言工具，是本次 PR 的核心价值所在。
- vllm/_aiter_ops.py（模块 算子封装；类别 source；类型 core-logic；符号 rms_norm, rms_norm2d_with_add）：为 rocm_aiter_ops 新增 rms_norm 与 rms_norm2d_with_add 两个静态方法，作为测试调用 AITER kernel 的统一入口，是测试套件能运行的前置条件。
- .buildkite/test-amd.yaml（模块 CI 配置；类别 config；类型 configuration）：为 mi355 镜像新增 Kernels Core Operation Test 门禁 job，以 parallelism 3 分片执行新建的算子测试，是 PR 的 CI 落地环节。

关键符号：rms_norm, rms_norm2d_with_add, test_rocm_aiter_rms_norm_vs_torch, test_rocm_aiter_rmsnorm_with_add_vs_torch, test_rocm_aiter_triton_rotary_embedding_vs_torch, _assert_close_budget, _assert_rel_error_quality

关键源码片段

tests/kernels/core/test_rocm_aiter_ops.py

新增 625 行 AITER 算子测试主体，覆盖 RMSNorm、残差融合、FP8 分组量化、RoPE 与端到端链路，并内置预算式数值断言工具，是本次 PR 的核心价值所在。

```
# tests/kernels/core/test_rocm_aiter_ops.py
# 预算式数值一致性断言：kernel 测试的核心工具。
```

```

# 与 torch.allclose 不同，它允许极少量元素超出容差（默认 pass_rate = 0.99999），
# 同时输出 max/mean/p99/p999/worst_ratio 等统计量，避免数值抖动造成 flaky CI。
def _assert_close_budget(
    actual: torch.Tensor,
    expected: torch.Tensor,
    *,
    label: str,
    atol: float,
    rtol: float = 0.0,
    pass_rate: float = 0.99999,
    max_violation_factor: float = 3.0,
) -> None:
    actual_f = actual.detach().float().flatten()
    expected_f = expected.detach().float().flatten()
    abs_diff = (actual_f - expected_f).abs()
    allowed = atol + rtol * expected_f.abs()

    total = abs_diff.numel()
    passed = int((abs_diff <= allowed).sum().item())
    failed = total - passed
    allowed_fail_rate = 1.0 - pass_rate
    max_abs = abs_diff.max().item()
    mean_abs = abs_diff.mean().item()
    p99_abs = _quantile(abs_diff, 0.99)
    p999_abs = _quantile(abs_diff, 0.999)
    worst_ratio = (abs_diff / allowed.clamp_min(1e-12)).max().item()
    max_atol = max_violation_factor * atol
    above_max_count = int((abs_diff > max_atol).sum().item())

    msg = (
        "[rocm_aiter_ops] "
        f"{label}: pass={passed / total:.4%} ({passed}/{total}) "
        f"fail={_format_observed_rate(failed, total)} "
        f"allowed_fail={_format_allowed_rate(allowed_fail_rate, total)} "
        f"atol={atol:g} rtol={rtol:g} "
        f"abs>{max_atol:g}={_format_observed_rate(above_max_count, total)} "
        f"allowed_above_max={_format_allowed_rate(0.0, total)} "
        f"max_abs={max_abs:.6g} mean_abs={mean_abs:.6g} "
        f"p99_abs={p99_abs:.6g} p999_abs={p999_abs:.6g} "
        f"worst_ratio={worst_ratio:.6g}"
    )
    # 失败率低于阈值时仅告警，保留可检索的统计信息供 CI 日志分析
    print(msg)
    if failed > 0:
        warnings.warn(msg, stacklevel=2)

    assert passed / total >= pass_rate, msg
    assert max_abs <= max_atol, msg
    assert mean_abs <= atol * 0.25, msg

```

vllm/_aiter_ops.py

为 rocm_aiter_ops 新增 rms_norm 与 rms_norm2d_with_add 两个静态方法，作为测试调用 AITER kernel 的统一入口，是测试套件能运行的前置条件。

```
# vllm/_aiter_ops.py 中 rocm_aiter_ops 类的两个新增静态方法
@staticmethod
def rms_norm(
    x: torch.Tensor,
    weight: torch.Tensor,
    epsilon: float,
) -> torch.Tensor:
    """RMSNorm via AITER kernel."""
    import aiter # 延迟导入，避免无 AITER 环境下模块导入失败

    return aiter.rms_norm(x, weight, epsilon)

@staticmethod
def rms_norm2d_with_add(
    x: torch.Tensor,
    residual: torch.Tensor,
    weight: torch.Tensor,
    epsilon: float,
) -> tuple[torch.Tensor, torch.Tensor]:
    """Fused residual-add + RMSNorm via AITER kernel.

    返回 (normalized_output, residual_sum) 两个张量；
    通过 aiter.rmsnorm2d_fwd_with_add 将归一化结果与残差和
    分别写入预分配的输出缓冲区，避免 kernel 内部额外分配。
    """
    import aiter

    out = torch.empty_like(x)
    residual_out = torch.empty_like(x)
    # 最后一个参数 0 为 layernorm 模式的标志位
    aiter.rmsnorm2d_fwd_with_add(out, x, residual, residual_out, weight, epsilon, 0)
    return out, residual_out
```

评论区精华

审核由 AndreasKaratzas 主导，围绕测试稳健性与数值语义展开，核心交锋如下：

- AITER 不可用时的处理策略：AndreasKaratzas 建议 'Probably best if we just skip if aiter is not found and supported (eg mi250 arch)', 避免在不受支持的架构上直接失败。作者采纳并实现 require_aiter 跳过机制。
- 容差数值合理性质疑：AndreasKaratzas 追问 'are the tolerances checked? are they numerically also correct?', 要求以 ULP 理论验证 FP16 容差。作者回应 'Updated to ULP based comparison and the default torch values', 最终 commit 8248cb1 将 RMSNorm 系列改为 bf16_ulp_distance 的 ULP 距离断言，规定最大 1 ULP。

- RoPE 参考实现的精度语义分歧（最核心讨论）：Rohan138 指出测试参考实现中 `t.float()` 会将操作数升到 FP32 再旋转，而 AITER Triton kernel 保持原生 bf16 dtype，两者并不一致；并进一步说明这在 vLLM 内部是已知分歧点——C++ 实现总是 upcast 到 FP32，而 Triton 实现默认保持原生 dtype（除非 `enable_fp32_compute`）。作者承认该问题并另开 PR 修复，同时以 `strict xfail` 挂起该测试。
- 上游问题追踪：AndreasKaratzas 要求为 RoPE 精度缺陷创建 AITER 上游 ticket 并链接到 xfail 的 reason 中，作者提供 <https://github.com/ROCm/aiter/issues/4765> 并写入测试注释。
 - AITER 不支持时跳过而非失败 (design): 作者实现 `require_aiter` 辅助函数，通过 `vllm._aiter_ops.is_aiter_found_and_supported` 判断并 `pytest.skip`。
 - 容差数值合理性与 ULP 比较 (correctness): 作者改用 `bf16_ulp_distance` 进行 ULP 距离断言，RMSNorm 系列规定最大 1 ULP，并保留预算统计辅助诊断。
 - RoPE 参考实现 FP32 upcast 与 AITER 原生 dtype 的分歧 (correctness): 作者承认分歧并另开 PR 修复参考实现，测试本身以 `strict xfail` 挂起并链接 `aiter#4765` 跟踪。
 - 为 RoPE 精度问题创建 AITER 上游 ticket (documentation): 作者提供 <https://github.com/ROCm/aiter/issues/4765> 并写入测试的 xfail reason。

风险与影响

- 风险：
 1. xfail 掩盖已知精度缺陷：test_rocm_aiter_triton_rotary_embedding_vs_torch 使用 `strict=True`，一旦 AITER 上游修复精度问题而本测试未及时同步，CI 会立即收到 XPASS failure，属于预期内的显式风险跟踪；但若长期无人跟进 `aiter#4765`，该测试会一直处于跳过状态，RoPE 回归信号实际缺失。
 2. 新增公开封装接口：vllm/_aiter_ops.py 的 `rms_norm`、`rms_norm2d_with_add` 虽仅服务测试，但作为 `rocm_aiter_ops` 的公开静态方法，后续若被生产路径误引用，需要额外维护 ABI 与行为契约。
 3. CI 并行分片负载：新 job 在 mi355 上以 `parallelism 3` 并行运行，且 `dependencies` 含 `csrc/` 全量，任何 kernel 内层改动都会触发该 job，可能拉长 mi355 队列等待时间；分片命令同时跑 `kernels/core`、`test_concat_mla_q.py`、`test_top_k_per_row.py` 三类测试，单 shard 失败定位仍需人工查看日志。
 4. 设备设置硬编码：fixture 中 `torch.set_default_device("cuda")` 依赖 ROCm 的 CUDA 语义映射，在纯 HIP 设备编号与 CUDA 编号不一致的异构环境中可能产生预期外行为。
 - 影响：对 ROCm 用户，AITER 相关 kernel (RMSNorm、FP8 量化、RoPE) 的回归将能在合并前被 CI 捕获，降低线上推理精度劣化风险；对开发团队，MI355 镜像新增一条算子级门禁路径，kernel 改动需要额外满足数值一致性预算，测试本身通过 ULP 与统计断言提供了可量化的通过标准；对 vLLM 架构，`rocm_aiter_ops` 新增的两个封装为将来在生产路径中直接复用 AITER RMSNorm 留下低摩擦入口。整体影响范围限定在 ROCm/CI 测试域，不触及推理主链路，影响程度中等偏下。
 - 风险标记：xfail 掩盖已知精度缺陷，新增 `rocm_aiter_ops` 公开封装接口，CI 并行分片增加队列负载，设备设置硬编码 CUDA 语义

关联脉络

- PR #44969 [ROCm][CI] Gating more ROCm tests: 同一 ROCm CI 测试覆盖扩展主题，此前通过扩大 gate 测试集提升 AMD CI 覆盖，本 PR 进一步在 MI355 镜像上新增算子级门禁，是其后续延续。
- PR #52763 [ROCM][CI] Attention test speedup: 并行推进的 ROCm 测试基建工作，通过解锁与提速注意力 kernel 测试改善 CI 效率，与本 PR 的算子测试门禁共同构成 ROCm 测试体系完善计划。
- PR #52822 [ROCm][CI] Add AMD CI Pull-Request Commands: AMD CI 基础设施配套，为 ROCm 相关 PR 提供独立 CI 触发与重试命令，本 PR 的 CI 门禁依赖该基础设施运行。