PR #50323 完整报告

vllm-project/vllm

[CI] Add option to raise an exception when NaNs are detected in logits

合并时间：2026-08-05 07:02

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/50323

执行摘要

- 一句话：新增 logits NaN fail-fast 开关，CI 全局检测
- 推荐动作：值得精读，但精读重点不是代码量（仅 +60/-3），而是其设计演进过程：从 per-eval 指标抓取到全局 fail-fast 环境变量的转变，体现了 CI 诊断能力的基础设施化思路。可以关注 raise_if_nan_logits 的实现细节（成功路径零分配、异常消息携带请求级信息）以及 env 变量间隐式联动的写法；对需要构建类似检测开关的团队有参考价值。

功能与动机

PR body 明确指出：logits 中的 NaN 往往与 KV cache 的 NaN 相伴出现，而注意力内核通过乘零进行 mask，一旦 KV cache 被 NaN 污染就会产生灾难性故障，属于推理服务的严重故障模式。关联的 Dao-AILab/flash-attention#1974 给出了具体复现：vLLM V1 中 hybrid 模型（如 attention + mamba2）的 mamba state 与 KV cache 共享底层内存但使用不同 dtype 视图，导致 KV 张量出现 bf16 下的 NaN 位模式，即使 FA3 理论上不应读取这些区域，输出仍会出错。本 PR 的目的是让这类问题在 CI 中立即暴露（fail-fast），而不是让 eval 静默产出错误结果。

实现拆解

1. 新增环境变量开关：在 vllm/envs.py 中定义 VLLM_RAISE_ON_LOGIT_NANS: bool = False，解析默认值为 0；同时让 VLLM_COMPUTE_NANS_IN_LOGITS 的解析逻辑 OR 上新开关，保证开启 fail-fast 时自动启用 NaN 计算，避免出现想抛异常却没算 NaN 的空转。
2. 提炼公共判定函数：在 vllm/v1/worker/utils.py 新增 raise_if_nan_logits(num_nans_in_logits)，传入请求级 NaN 计数映射；全零则直接返回，否则构造 corrupted_requests 字典并抛出 RuntimeError，消息中直接列出被污染的请求 ID 与 NaN 数量。
3. 同步执行路径接入：在 vllm/v1/worker/gpu_model_runner.py 的 _get_nans_in_logits 返回统计前，按 envs.VLLM_RAISE_ON_LOGIT_NANS 决定是否调用 raise_if_nan_logits，覆盖 v1/v2 两种模型执行器。
4. 异步输出路径接入：在 vllm/v1/worker/gpu/async_utils.py 的 AsyncOutput.get_output 中，在组装 num_nans_in_logits 映射后同样按环境变量调用 raise_if_nan_logits，保证 CUDA graph 重叠执行场景下也能及时暴露问题。

5. 测试与 CI 配套：tests/basic_correctness/test_basic_correctness.py 新增 test_raise_on_logit_nans，通过 monkeypatch 在 compute_logits 中注入单个 NaN，分别以 v1/v2 model runner 跑真实推理并断言 RuntimeError；tests/v1/worker/test_gpu_model_runner.py 的既有单测显式关闭新开关以保持统计逻辑可独立测试。全局开启由配套仓库 vllm-project/ci-infra#444 在 buildkite 上统一设置，本仓库不硬编码。

关键文件：

- vllm/v1/worker/utils.py（模块 诊断工具；类别 source；类型 core-logic；符号 raise_if_nan_logits）：新增 raise_if_nan_logits 工具函数，是整个 fail-fast 机制的判定与异常构造核心，被同步与异步两条路径复用。
- vllm/v1/worker/gpu_model_runner.py（模块 模型执行；类别 source；类型 dependency-wiring；符号 _get_nans_in_logits）：同步执行路径的接入点，_get_nans_in_logits 在返回统计前根据 env 决定是否抛异常，影响 v1/v2 model runner。
- vllm/envs.py（模块 环境配置；类别 source；类型 configuration；符号 VLLM_RAISE_ON_LOGIT_NANS, VLLM_COMPUTE_NANS_IN_LOGITS）：新增 VLLM_RAISE_ON_LOGIT_NANS 环境变量，并让 VLLM_COMPUTE_NANS_IN_LOGITS 隐式随其开启；默认值最终恢复为 0，避免影响生产。
- vllm/v1/worker/gpu/async_utils.py（模块 异步输出；类别 source；类型 dependency-wiring；符号 AsyncOutput.get_output）：异步输出路径 AsyncOutput.get_output 同样接入 fail-fast，保证重叠执行场景下也能及时暴露 NaN。
- tests/basic_correctness/test_basic_correctness.py（模块 正确性测试；类别 test；类型 test-coverage；符号 test_raise_on_logit_nans, compute_nan_logits）：端到端验证 fail-fast：在 v1/v2 两种 model runner 下注入 NaN 并断言 RuntimeError，是行为契约的关键测试。
- tests/v1/worker/test_gpu_model_runner.py（模块 runner 测试；类别 test；类型 test-coverage；符号 test_get_nans_in_logits）：原有 _get_nans_in_logits 单测需要显式关闭新 env，避免被全局开关干扰，保证统计逻辑独立可测。

关键符号：raise_if_nan_logits, _get_nans_in_logits, AsyncOutput.get_output, test_raise_on_logit_nans, compute_nan_logits, test_get_nans_in_logits

关键源码片段

vllm/v1/worker/utils.py

新增 raise_if_nan_logits 工具函数，是整个 fail-fast 机制的判定与异常构造核心，被同步与异步两条路径复用。

```
def raise_if_nan_logits(num_nans_in_logits: Mapping[str, int]) -> None:
    # 任一请求出现 NaN 即触发异常；全为 0 时直接返回，避免无谓分配。
    if not any(num_nans_in_logits.values()):
        return

    # 仅收集真正被污染的请求，便于快速定位故障来源。
    corrupted_requests = {
```

```
            req_id: num_nans
            for req_id, num_nans in num_nans_in_logits.items()
            if num_nans > 0
        }
        # RuntimeError 会让上层测试 / 服务直接失败，而不是静默产出乱码。
        raise RuntimeError(f'NaNs detected in logits: {corrupted_requests}')
```

vllm/v1/worker/gpu_model_runner.py

同步执行路径的接入点，_get_nans_in_logits 在返回统计前根据 env 决定是否抛异常，影响 v1/v2 model runner。

```
def _get_nans_in_logits(
    self,
    logits: torch.Tensor | None,
) -> dict[str, int]:
    try:
        if logits is None:
            return {req_id: 0 for req_id in self.input_batch.req_ids}

        num_nans_in_logits = {}
        # 按词表维度统计每行 NaN 数量，并同步到 CPU 便于逐请求换算。
        num_nans_for_index = logits.isnan().sum(dim=-1).cpu().numpy()
        for req_id in self.input_batch.req_ids:
            req_index = self.input_batch.req_id_to_index[req_id]
            num_nans_in_logits[req_id] = (
                int(num_nans_for_index[req_index])
                if num_nans_for_index is not None and req_index < logits.shape[0]
                else 0
            )
        # 仅在显式开启 fail-fast 时抛出异常；默认保持向后兼容。
        if envs.VLLM_RAISE_ON_LOGIT_NANS:
            raise_if_nan_logits(num_nans_in_logits)
        return num_nans_in_logits
    except IndexError:
        # 竞态下请求可能已被移除，静默返回空字典。
        return {}
```

vllm/envs.py

新增 VLLM_RAISE_ON_LOGIT_NANS 环境变量，并让 VLLM_COMPUTE_NANS_IN_LOGITS 隐式随其开启；默认值最终恢复为 0，避免影响生产。

```
# 开启 logits NaN 检测（可能增加计算开销），主要用于排查底层 bug 或坏硬件。
'VLLM_COMPUTE_NANS_IN_LOGITS': lambda: bool(
    int(os.getenv('VLLM_COMPUTE_NANS_IN_LOGITS', '0'))
    or int(os.getenv('VLLM_RAISE_ON_LOGIT_NANS', '0'))
),
# logits 中出现 NaN 时直接抛异常；开启后隐式启用上方 NaN 计算。
'VLLM_RAISE_ON_LOGIT_NANS': lambda: bool(
    int(os.getenv('VLLM_RAISE_ON_LOGIT_NANS', '0'))
```

),

评论区精华

核心讨论围绕检测方式展开：

- njhill 在 issue 评论中建议：与其靠抓取 metrics，不如让 VLLM_COMPUTE_NANS_IN_LOGITS 模式在遇到 NaN 时抛异常，然后全局在 CI 设置，*doesn't even need to be added per test. Basically it would just blow up the test if encountered*。
- mgoin 附议并强调希望全局开启、让 pytest/script 直接失败。
- 作者据此将方案从逐个 *eval* 抓取 vllm:corrupted_requests_total 指标改为新增 VLLM_RAISE_ON_LOGIT_NANS 运行时开关，并提到配套的 *ci-infra#444* 会在 buildkite 全局启用。
- AndreasKaratzas 在 *deepseek_v2_lite_ep_eplb.sh* 的 diff 上询问是否由各硬件后端决定开关，或直接在 AMD 的 run-am-tests.sh 默认开启；最终该文件在方案调整后未保留逐 *eval* 设置，全局由 *ci-infra* 控制。
- *depthfirst-app* bot 发现中间提交把 os.getenv 默认值写成 1，与类注解 False 矛盾，会让所有生产实例在 NaN 时直接崩溃；作者在收尾提交中恢复默认 0。
- 全局 fail-fast 替代逐 *eval* 指标抓取 (design): 作者采纳建议，新增 VLLM_RAISE_ON_LOGIT_NANS 开关，移除逐 *eval* 的 metrics 抓取与辅助函数，并在 *ci-infra#444* 中全局启用。
- 是否由各硬件后端统一默认开启 (question): 最终该逐 *eval* 设置文件未保留，vLLM 仓库内保持 opt-in；是否在 AMD 默认开启未明确，全局由 *ci-infra* 控制。
- 环境变量默认值一度为 1 的回归风险 (correctness): 作者在收尾提交 Fix NaN *detection test cleanup* 中恢复默认 0，保持 opt-in。

风险与影响

- 风险：
  1. 性能开销：开启后每个前向步都要对 logits 做 isnan().sum(dim=-1) 并执行 .cpu() 同步以逐请求统计，会打断 GPU 流水线；这是设计上的已知取舍，因此默认关闭，仅 CI/ 调试时开启。
  2. 生产误开风险：中间提交曾把默认值设为 1，若按那个版本发布，任何一次 NaN logits 都会让整个服务抛 RuntimeError 崩溃；最终已恢复 opt-in 默认 0，但后续修改 *env* 解析时需警惕再次引入同类回归。
  3. 异常语义变化：_get_nans_in_logits 捕获 IndexError 后静默返回空字典，若请求在统计期间被移除，NaN 会被吞掉；fail-fast 模式下该静默路径可能掩盖真实问题。
  4. 测试耦合：端到端测试通过 monkeypatch 注入 NaN 并匹配异常消息 NaNs *detected* in logits，若未来修改消息文案或 *compute_logits* 签名，测试需要同步维护。
- 影响：

1. 用户：默认零影响（新变量默认关闭）；显式开启后，logits 出现 NaN 会从静默生成乱码变为明确报错，大幅降低 FA3 类内核问题的定位成本。
2. 系统：为 CI 提供全局 fail-fast 能力，配合 ci-infra#444 可在 buildkite 全量测试中第一时间捕获 KV *cache*/ 注意力核污染，避免 *eval* 用垃圾数字通过。
3. 团队：新增一个环境变量和公共工具函数，后续其他执行路径（如 pooling、CPU runner）若要接入只需复用 raise_if_nan_logits；需要同步更新环境变量文档。 - 风险标记：默认关闭，生产无影响，开启时有 GPU-CPU 同步开销，异常消息被测试断言耦合，IndexError 静默吞掉 NaN，中间版本曾默认开启

关联脉络

- 暂无明显关联 PR