PR #26521 完整报告

sgl-project/sglang

fix: copy seq_lens in TRTLLM MHA draft decode cuda graph capture

合并时间：2026-05-29 12:55

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26521

执行摘要

- 一句话：修复 TRTLLM MHA draft decode CUDA graph 捕获时 cache_seqlens 未拷贝
- 推荐动作：建议精读 PR 的 diff 和 PR body，理解 CUDA graph 捕获时 draft decode 分支的 metadata 初始化逻辑。这是一个典型的捕获路径与 replay 路径不一致导致的 bug，值得学习。

功能与动机

在 TRTLLM MHA 注意力后端中，draft decode 分支的 CUDA graph 捕获使用了一个预分配、零初始化的 cache_seqlens 缓冲区，但未将当前 seq_lens 拷贝进去。结果捕获的 kernel 运行时 cache_seqlens 为 0（空 KV 范围），导致 softmax 归一化产生 NaN。该问题在 EAGLE3 + trtllm_mha + SGLANG_SPEC_NAN_DETECTION=1 时可稳定复现。

实现拆解

在 python/sglang/srt/layers/attention/trtllm_mha_backend.py 的 init_forward_metadata_capture_cuda_graph 方法中，draft decode 分支（spec_info is not None）内，在绑定 metadata.cache_seqlens_int32 到预分配缓冲区之后，增加一行 metadata.cache_seqlens_int32.copy_(seq_lens + self.speculative_step_id + 1)，将当前序列长度加上推测步数后的值拷贝到缓冲区中。该操作与 replay 路径中的显式 copy_ 行为一致。

关键文件：

- python/sglang/srt/layers/attention/trtllm_mha_backend.py（模块 注意力层；类别 source；类型 core-logic；符号 init_forward_metadata_capture_cuda_graph）：核心修复文件，在 draft decode 的 CUDA graph 捕获分支中增加 cache_seqlens 拷贝。

关键符号：init_forward_metadata_capture_cuda_graph

关键源码片段

python/sglang/srt/layers/attention/trtllm_mha_backend.py

核心修复文件，在 draft decode 的 CUDA graph 捕获分支中增加 cache_seqlens 拷贝。

```
# python/sglang/srt/layers/attention/trtllm_mha_backend.py
# 在 init_forward_metadata_capture_cuda_graph 方法的 draft decode 分支中
# 修复前：metadata.cache_seqlens_int32 绑定到预分配的零初始化缓冲区，但未写入当前值
```

```
# 修复后：增加 copy_ 调用，使捕获的 CUDA graph 使用正确的序列长度
if spec_info is not None:
    # Draft Decode ( 仅支持 topk = 1)
    metadata.cache_seqlens_int32 = self.decode_cuda_graph_metadata[
        "cache_seqlens"
    ][:bs]
    # === 修复：将当前 seq_lens 加上推测步数后拷贝到预分配缓冲区 ===
    metadata.cache_seqlens_int32.copy_(
        seq_lens + self.speculative_step_id + 1
    )
    # === 后续计算使用正确的 cache_seqlens ===
    metadata.max_seq_len_k = seq_lens.max().item() + (
        self.speculative_step_id + 1
    )
    metadata.cu_seqlens_q = self.decode_cuda_graph_metadata["cu_seqlens_q"][: bs + 1]
    metadata.cu_seqlens_k = torch.nn.functional.pad(
        torch.cumsum(
            metadata.cache_seqlens_int32, dim=0, dtype=torch.int32
        ),
        (1, 0),
    )
    # ... 后续 page_table 和 swa_page_table 绑定不变 ...
```

评论区精华

无 review 评论。PR body 中作者 libertyeagle 详细描述了问题的复现步骤、根因分析（软 max 在空 KV 范围时产生 NaN）以及修复方案。验证表明修复后 /health 返回 200，/generate 输出与预期一致，SGLANG_SPEC_NAN_DETECTION=1 不再触发 NaN。

- 暂无高价值评论线程

风险与影响

- 风险：风险极低。改动手法仅为在捕获路径中增加一个已被 replay 路径验证过的 copy_ 操作，且变更量仅 3 行。但本次改动未附带专门针对该场景的单元测试，未来重构时可能被遗漏。
- 影响：影响范围限定于使用 TRTLLM MHA 后端 + speculative decoding (如 EAGLE3) + CUDA graph 捕获的场景。修复前该组合会导致 NaN 错误，修复后可正常工作。对其他后端或路径无影响。
- 风险标记：缺少测试覆盖

关联脉络

- PR #26655 Fix TRTLLM MHA draft decode cache seqlens replay: 同一个文件、同一个模块的类似修复，涉及 cache seqlens 的 replay 路径