PR #31312 完整报告

sgl-project/sglang

Fix LongCat n-gram token-table crashes on padded batches

合并时间：2026-07-22 20:24

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/31312

执行摘要

- 一句话：修复 LongCat n-gram 在填充批次上的崩溃
- 推荐动作：该 PR 值得合并，但建议后续补充单元测试覆盖填充批次场景，以及采纳 Gemini 建议添加 _ng_bs 守卫到 update_ngram_token_table_after_sampling 中以增强鲁棒性。

功能与动机

在 cuda-graph 或 EP eager 填充模式下，seq_lens、next_token_ids、req_pool_indices 被填充到 graph batch size，而 batch_size 是真实请求数，导致逐请求张量索引不一致，进而触发 RuntimeError: The expanded size of the tensor (N) must match the existing size (M) 和 cudaErrorIllegalAddress 崩溃。

实现拆解

1. 修复 update_ngram_token_table_after_sampling（ngram_embedding_manager.py）：将 seq_lens、next_token_ids、req_pool_indices 及输出张量切片至 batch_size，确保只有真实请求更新 token table，填充行不会污染表格。
2. 修复 NgramEmbedding.forward（layers/n_gram_embedding.py）：引入 _ng_bs 守卫变量，取 min(batch_size, req_lens.shape[0], column_starts.shape[0])，驱动 cumsum 和 n-gram kernel 的请求循环，防止 kernel 越界读取 column_starts/req_lens。
3. 测试验证：在 meituan-longcat/LongCat-2.0-FP8、tp16/ep16 over EFA 上验证，并发数 1-64 下修复后不再崩溃且输出正确。

关键文件：

- python/sglang/srt/model_executor/model_runner_components/ngram_embedding_manager.py（模块 嵌入管理；类别 source；类型 data-contract；符号 update_ngram_token_table_after_sampling）：修复了更新 token table 时未对 padded 张量切片的问题，是导致 RuntimeError 的直接原因。
- python/sglang/srt/layers/n_gram_embedding.py（模块 嵌入层；类别 source；类型 core-logic；符号 NgramEmbedding.forward）：修复了 n-gram kernel 读取越界的问题，通过引入 _ng_bs 守卫确保索引不超出数组范围。

关键符号：update_ngram_token_table_after_sampling, NgramEmbedding.forward

关键源码片段

python/sglang/srt/model_executor/model_runner_components/ngram_embedding_manager.py

修复了更新 token table 时未对 padded 张量切片的问题，是导致 RuntimeError 的直接原因。

```
# 文件：python/sglang/srt/model_executor/model_runner_components/ngram_embedding_
manager.py
# 函数：update_ngram_token_table_after_sampling

def update_ngram_token_table_after_sampling(
    *,
    ngram_embedding_info,
    next_token_ids: torch.Tensor,
    req_pool_indices: torch.Tensor,
    seq_lens: torch.Tensor,
    batch_size: int,
) -> bool:
    """使用采样后的 token 更新 ngram token table。"""
    skip_token_table_update = ngram_embedding_info.skip_token_table_update
    if skip_token_table_update is not None:
        # 跳过未完成预填充的请求
        indices = (~skip_token_table_update).nonzero(as_tuple=True)[0]
        if indices.numel() == 0:
            return False
        update_token_table(
            ne_token_table=ngram_embedding_info.token_table,
            tokens=next_token_ids[indices].to(torch.int32),
            row_indices=req_pool_indices[indices],
            column_starts=seq_lens[indices].to(torch.int32),
            req_lens=torch.ones(indices.numel(), dtype=torch.int32, device=next_token_ids.device),
            ignore_tokens=None,
        )
        return True

    # 修复：seq_lens / next_token_ids / req_pool_indices 可能被填充到 cuda-graph
    # 的 batch size，而 batch_size 是真实请求数。切片到 batch_size 可防止
    # 填充行污染 token table（同时确保形状匹配）。
    ngram_embedding_info.out_column_starts[:batch_size] = seq_lens[:batch_size]
    ngram_embedding_info.out_req_lens[:batch_size] = 1
    update_token_table(
        ne_token_table=ngram_embedding_info.token_table,
        tokens=next_token_ids[:batch_size].to(torch.int32),
        row_indices=req_pool_indices[:batch_size],
        column_starts=ngram_embedding_info.out_column_starts[:batch_size],
        req_lens=ngram_embedding_info.out_req_lens[:batch_size],
        ignore_tokens=None,
    )
```

```
    return True
```

python/sglang/srt/layers/n_gram_embedding.py

修复了 n-gram kernel 读取越界的问题，通过引入 _ng_bs 守卫确保索引不超出数组范围。

```
# 文件：python/sglang/srt/layers/n_gram_embedding.py
# 类：NgramEmbedding

def forward(self, input_ids: torch.Tensor, forward_batch: ForwardBatch):
    if (
        forward_batch.forward_mode.is_extend()
        or forward_batch.forward_mode.is_decode()
    ):
        ngram_embedding_info = forward_batch.ngram_embedding_info
        # 修复：ngram_info 数组可能比 forward_batch.batch_size 短
        # ( 混合 / 重叠批次 )。基于数组长度驱动请求循环，防止 kernel 越界
        # 读取 column_starts / req_lens (cudaErrorIllegalAddress)。
        _ng_bs = min(
            forward_batch.batch_size,
            ngram_embedding_info.req_lens.shape[0],
            ngram_embedding_info.column_starts.shape[0],
        )
        torch.cumsum(
            ngram_embedding_info.req_lens[:_ng_bs],
            dim=0,
            dtype=torch.int32,
            out=self.exclusive_req_len_sums[1 : 1 + _ng_bs],
        )
        compute_n_gram_ids(
            ne_n=self.over_embedding_n,
            ne_k=self.over_embedding_k,
            ne_weights=self.oe_weights,
            ne_mods=self.oe_mods,
            tokens=input_ids.to(torch.int32),
            exclusive_ne_embedder_size_sums=self.exclusive_oe_embedder_size_sums,
            exclusive_req_len_sums=self.exclusive_req_len_sums[: _ng_bs + 1],
            ne_token_table=ngram_embedding_info.token_table,
            row_indices=forward_batch.req_pool_indices[:_ng_bs],
            column_starts=ngram_embedding_info.column_starts[:_ng_bs],
            n_gram_ids=self.oe_n_gram_ids[: len(input_ids)],
            eos_token_id=self.eos_token_id,
        )
    # ... 后续代码不变
```

评论区精华

Gemini Code Assist 机器人建议在 update_ngram_token_table_after_sampling 中增加与 NgramEmbedding.forward 类似的 _ng_bs 守卫，以防止 batch_size 超过输出张量分配大小导致越界。该建议未在后续讨论中被接受或实施，但 PR 作者未进一步回应，审阅者 BBuf 已

批准该 PR。

- 在 update_ngram_token_table_after_sampling 中添加 _ng_bs 守卫 (correctness): 未采纳：PR 作者未回应，但审阅者 BBuf 已批准 PR。

风险与影响

- 风险：最低风险：修复范围明确（仅 LongCat n-gram 路径），改动量小（+22/-12 行），且已在真实模型上验证。但未添加单元测试，回归依赖于现有集成测试。
- 影响：影响范围较小：仅影响使用 LongCat-2.0 n-gram embedding 并且在 cuda-graph 或 EP eager 模式下填充批次的用户。修复后这些用户将避免 decode 崩溃，输出正确。
- 风险标记：缺少测试覆盖

关联脉络

- PR #32015 [Kernel] Phase 4 batch-2: migrate JIT operator groups into kernels.ops (no shims) (RFC #29630): 同属 LongCat/n-gram 相关功能模块，涉及 n-gram kernel 的迁移重构。