PR #6267 完整报告

verl-project/verl

[megatron] fix: fix bugs when using position_ids in cp

合并时间：2026-05-09 12:27

原文链接：http://prhub.com.cn/verl-project/verl/pull/6267

执行摘要

- 一句话：修复 Megatron CP 中 position_ids 计算错误
- 推荐动作：建议精读本 PR，尤其是 position_ids 计算公式的推导（PR body 中有详细对比）。对于代码审查者，重点关注 model_forward.py 中条件修改的潜在影响，建议后续添加 VLM+MTP 的 CI 测试。已获 wuxibin89 approve，可合入。

功能与动机

PR #5561 为支持 MTP（Multi-Token Prediction）引入了显式 position_ids 传递，但在 CP 场景下未充分测试。PR body 明确指出两个 bug：

1) cp=2 时 position_ids 错了一位； 2) cp=2 时 rank=1 的后半段 position_ids 仍然错误。PR body 还详细分析了根因：input_ids 使用 remain_start 作为数据索引（正确），但 position_ids 使用了 seqlen_padded_i - remain_len 作为起点，该公式仅对 cp_rank=0（最后一段）成立，对 cp_rank=1（中间段）错误。

实现拆解

1. 修正 position_ids 计算公式（verl/models/mcore/util.py）：
   - 新增变量 seqlen_orig_i = seqlens_in_batch_cpu[i]，记录填充前的原始序列长度，用于后续 clamp 操作。
   - 将剩余块的 position_ids 生成从 torch.arange(seqlen_padded_i - remain_len, seqlen_padded_i) 改为 torch.arange(remain_start, pos_end)，其中 pos_end = min(remain_end, seqlen_orig_i)。
   - valid_pos_len = pos_end - remain_start 确保只填充有效位置区间，padding 区域自动跳过。
   - 此改动确保了 position_ids 在 CP 分片后的连续性：对于 cp_rank=0，position_ids 为 [0, half_seqlen) 和 [half_seqlen, seqlen_orig_i)；对于 cp_rank=1，则正确对应 [half_seqlen, 2half_seqlen) 和 [2half_seqlen, seqlen_orig_i + half_seqlen) 等。
2. 限制 position_ids 传递条件（verl/models/mcore/model_forward.py）：
   - 将 model() 调用时的 position_ids=position_ids_rmpad if not vision_model else None 改为 position_ids=position_ids_rmpad if mtp_enable_train else None。
   - 理由：mcore 在无 MTP 时可通过 packed_seq_params 内部计算 position_ids，显式传递破坏了原始行为（PR #5561 之前此处为 None）。Vision 模型也移除了位置传递，因

为 VLM 内部会自行计算 position_ids。

- 此改动降低了非 MTP 场景下的回归风险，符合讨论中 ISEEKYAN 和 wuxibin89 的建议。

3. 测试配套：本次改动未包含测试文件变更，CI 中 PR #5561 引入的 MTP 计算路径缺乏覆盖。

关键文件：

- verl/models/mcore/util.py（模块 模型；类别 source；类型 data-contract）：核心修复文件。修改 zigzag 切分下 position_ids 计算逻辑，将第二块的起始位置从 seqlen_padded 改为 remain_start，并 clamp 到原始序列长度。
- verl/models/mcore/model_forward.py（模块 模型；类别 source；类型 data-contract）：修改 position_ids 传递条件，避免非 MTP 场景下不必要的显式传递，降低回归风险。

关键符号：preprocess_thd_engine

关键源码片段

verl/models/mcore/util.py

核心修复文件。修改 zigzag 切分下 position_ids 计算逻辑，将第二块的起始位置从 seqlen_padded 改为 remain_start，并 clamp 到原始序列长度。

```
# verl/models/mcore/util.py ( 变化片段 )

seqlen_padded_i = seqlens_in_batch_padded_cpu[i]
# 新增：记录原始序列长度（不含 padding），用于后续 clamp
seqlen_orig_i = seqlens_in_batch_cpu[i]
seqlen = seqlen_padded_i // cp_size
half_seqlen = seqlen // 2
start_idx = cu_seqlens_padded_cpu[i] // cp_size

# 处理第一块 (chunk 0): position_ids 直接使用 cp_rank 对应的偏移
position_ids_rmpad[start_idx : start_idx + half_seqlen] = torch.arange(
    half_seqlen * cp_rank, half_seqlen * (cp_rank + 1), dtype=torch.long, device=input_ids.device
)

remain_start = seqlen_padded_i - half_seqlen * (cp_rank + 1)
remain_end = seqlen_padded_i - half_seqlen * cp_rank
remain_end = min(remain_end, d.shape[0])
remain_len = remain_end - remain_start
if remain_len > 0:
    input_ids_rmpad[start_idx + half_seqlen : start_idx + half_seqlen + remain_len] = d[
        remain_start:remain_end
    ]
    # 修复：使用 remain_start 作为 position_ids 起始值，并 clamp 到原始序列长度
    pos_end = min(remain_end, seqlen_orig_i)
    valid_pos_len = pos_end - remain_start
    if valid_pos_len > 0:
        position_ids_rmpad[start_idx + half_seqlen : start_idx + half_seqlen + valid_pos_len] = (
            torch.arange(remain_start, pos_end, dtype=torch.long, device=input_ids.device)
```

)
```

verl/models/mcore/model_forward.py

修改 position_ids 传递条件，避免非 MTP 场景下不必要的显式传递，降低回归风险。

```
# verl/models/mcore/model_forward.py ( 变化片段 )

output_orig = model(
    input_ids=input_ids_rmpad,
    attention_mask=attention_mask,
    # 仅当 MTP 启用时才显式传递 position_ids;
    # 否则 mcore 会通过 packed_seq_params 内部计算，与 #5561 前行为一致
    position_ids=position_ids_rmpad if mtp_enable_train else None,
    packed_seq_params=packed_seq_params,
    **model_kwargs,
)
```

评论区精华

1. 是否应限制 position_ids 传递范围（ISEEKYAN 提出）：
   - ISEEKYAN 指出，在 #5561 之前此路径使用 position_ids=None，显式传递 position_ids 会改变非 MTP 场景下的 THD 行为，建议仅在 MTP 路径或使用 mtp_enable_train 门控时传递。
   - wuxibin89 进一步询问 VLM+MTP 场景是否需要显式传递。
   - ArronHZG 和 wuxibin89 共同决策：修改为 if mtp_enable_train else None，只对 MTP 场景传递 position_ids，VLM 场景由 VLM 内部计算。
2. CI 覆盖缺失：ArronHZG 要求补充 MTP 计算的 CI 测试，但本 PR 未实现，可能需在后续 PR 中跟进。

- 限制 position_ids 传递范围，避免影响非 MTP 场景 (design): 改为仅当 mtp_enable_train 时传递 position_ids，VLM 场景由内部计算。
- 缺少 MTP 相关 CI 测试 (testing): 本 PR 未添加测试，需后续 PR 跟进。

风险与影响

- 风险：
  1. 回归风险（中）： - model_forward.py 中 position_ids 传递条件的修改（not vision_model → mtp_enable_train）可能影响多模态 VLM+MTP 组合场景。Review 中 wuxibin89 提出了这一问题，但未达成明确结论。若 VLM+MTP 需要显式 position_ids，当前修改可能导致其缺失。 - 非 MTP + CP 场景此前未测试，修复后的行为虽符合预期，但缺乏回归测试覆盖。
1. Padding 边界风险（低）：pos_end = min(remain_end, seqlen_orig_i) 假设 padding 仅在序列末尾，如果未来对齐策略变化（如中间填充），此逻辑需重新评估。
2. 性能影响（无）：改动仅涉及 CPU 端的整数运算和少量条件判断，不会影响计算密集型操作。 - 影响：
```

- 影响范围：Megatron 引擎中启用上下文并行（CP > 1）且使用 THD 格式的所有训练流程，包括 MTP 场景。
- 严重程度：CP 场景下 position_ids 错误会导致训练无法收敛或 loss 异常，属于功能性 bug。修复后 CP 下的 position_ids 语义正确，确保序列位置建模准确性。
- 用户影响：仅影响使用 CP 且通过 #5561 启用 MTP 的用户，对于非 MTP/ 非 CP 场景无影响。
- 风险标记：核心路径变更，缺少测试覆盖

关联脉络

- PR #5561 [megatron] feat: support MTP in THD format: 本 PR 修复了 #5561 引入的 position_ids 在 CP 场景下的两个 bug。