PR #6345 完整报告

verl-project/verl

[fsdp] fix: build no-padding attention mask from input ids

合并时间：2026-05-14 20:53

原文链接：http://prhub.com.cn/verl-project/verl/pull/6345

执行摘要

- 一句话：修复 FSDP NO_PADDING 路径下 attention mask 使用 response-only loss_mask 的 bug
- 推荐动作：建议精读。该 PR 清晰展示了如何通过 input_ids.offsets().diff() 从 jagged tensor 获取序列长度并正确构建 attention mask，测试设计也值得借鉴（显式对比旧行为与新行为，并验证下游形状契约）。

功能与动机

Issue #6278 报告当 use_remove_padding=False 且 strategy=fsdp2 时，KL/clipfrac 异常高。定位到原因：attention mask 从 response-only 的 loss_mask 构建，但被用作完整 prompt+response 序列的 mask。PR 修复此问题，使 attention mask 覆盖完整序列。

实现拆解

1. 新增核心辅助函数build_attention_mask_from_nested（verl/workers/utils/padding.py）：利用嵌套张量 input_ids.offsets().diff() 计算每条序列的真实长度，然后通过广播比较位置索引与序列长度，高效生成形状为 (batch_size, max_seq_len) 的 int32 attention mask。
2. 替换 FSDP 引擎中的 mask 构造（verl/workers/engine/fsdp/transformer_impl.py）：在 prepare_model_inputs 方法的 NO_PADDING 且 use_remove_padding=False 路径下，移除原先基于 loss_mask 的构造逻辑，改为调用 build_attention_mask_from_nested 并传入 micro_batch["input_ids"]。不再需要 loss_mask 参数，同时将 max_seq_len 类型显式转换为 int。
3. 添加 CPU 单元测试（tests/utils/test_padding_on_cpu.py）：新增 test_build_attention_mask_from_nested_uses_full_sequence_lengths，验证输入不同长度序列时 mask 的正确性。
4. 添加 GPU 回归测试（tests/models/test_fsdp_no_padding_on_gpu.py）：新增完整测试文件，包含两个测试用例：
   - test_prepare_model_inputs_uses_full_sequence_attention_mask_on_gpu：对比旧行为（response-only mask）与新行为（full-sequence mask），确认修复有效。
   - test_prepare_model_outputs_can_be_sliced_back_to_response_shape_on_gpu：验证下游 prepare_model_outputs 返回的嵌套 log_probs 能正确通过 no_padding_2_padding 切回响应形状。

关键文件：

- verl/workers/utils/padding.py（模块 工具函数；类别 source；类型 core-logic；符号 build_attention_mask_from_nested）：新增核心函数 build_attention_mask_from_nested，是整个修复的算法基础。
- verl/workers/engine/fsdp/transformer_impl.py（模块 FSDP 引擎；类别 source；类型 dependency-wiring）：在 FSDP 引擎的 prepare_model_inputs 方法中替换旧的 mask 构造逻辑，是 bug 的实际修复点。
- tests/models/test_fsdp_no_padding_on_gpu.py（模块 GPU 测试；类别 test；类型 test-coverage；符号 _nested, _make_micro_batch, _fsdp_engine_with_lm_head_cls, _legacy_attention_mask_from_response_loss_mask）：新增 GPU 回归测试，显式验证旧行为错误和新行为正确，并覆盖下游形状契约，是修复质量的关键保障。
- tests/utils/test_padding_on_cpu.py（模块 CPU 测试；类别 test；类型 test-coverage；符号 test_build_attention_mask_from_nested_uses_full_sequence_lengths）：新增 CPU 单元测试，确保 build_attention_mask_from_nested 函数在不同输入下的行为正确。

关键符号：build_attention_mask_from_nested, prepare_model_inputs

关键源码片段

verl/workers/utils/padding.py

新增核心函数 build_attention_mask_from_nested，是整个修复的算法基础。

```
def build_attention_mask_from_nested(input_ids: torch.Tensor, max_seq_len: int | None = None)
-> torch.Tensor:
    """从嵌套的 input_ids 构建填充后的完整序列 attention mask。

    Args:
        input_ids: jagged layout 的嵌套张量，每条样本的 id 连续存储。
        max_seq_len: 显式指定最大序列长度；若为 None 则使用 batch 内的最大长度。

    Returns:
        形状为 (batch_size, max_seq_len) 的 int32 张量，1 表示有效位置，0 表示填充。
    """
    assert input_ids.is_nested, "input_ids 必须是嵌套张量"
    device = input_ids.values().device
    # 通过 offsets 的 diff 计算每条序列的真实长度
    seq_lens = input_ids.offsets().diff().to(device=device)
    if max_seq_len is None:
        max_seq_len = int(seq_lens.max().item())
    # 生成位置索引并广播比较，高效构造 mask
    positions = torch.arange(max_seq_len, device=device).unsqueeze(0)
    return (positions < seq_lens.unsqueeze(1)).to(torch.int32)
```

verl/workers/engine/fsdp/transformer_impl.py

在 FSDP 引擎的 prepare_model_inputs 方法中替换旧的 mask 构造逻辑，是 bug 的实际修复点。

```
# 在 prepare_model_inputs 中，当 use_remove_padding=False 且 pad_mode == NO_PADDING
时：
if pad_mode == DatasetPadMode.NO_PADDING:
    input_ids = micro_batch["input_ids"]
    position_ids = micro_batch["position_ids"]
    pad_token_id = tu.get_non_tensor_data(data=micro_batch, key="pad_token_id", default=0)
    batch_size = micro_batch.batch_size[0]
    seq_len_effective = input_ids.offsets().diff()
    max_seq_len = int(seq_len_effective.max().item())

    # 填充 input_ids 和 position_ids 到 dense
    input_ids = torch.nested.to_padded_tensor(
        input_ids, padding=pad_token_id, output_size=(batch_size, max_seq_len)
    )
    if position_ids.dim() == 3:
        position_ids = torch.nested.to_padded_tensor(
            position_ids, padding=0, output_size=(batch_size, 4, max_seq_len)
        ).transpose(0, 1)
    else:
        position_ids = torch.nested.to_padded_tensor(
            position_ids, padding=0, output_size=(batch_size, max_seq_len)
        )

    # 关键修复：使用完整的 input_ids 而非 loss_mask 构造 attention mask
    attention_mask = build_attention_mask_from_nested(
        input_ids=micro_batch["input_ids"], max_seq_len=max_seq_len
    )

    model_inputs = {
        "input_ids": input_ids,
        "attention_mask": attention_mask,
        "position_ids": position_ids,
    }
```

评论区精华

无实质性讨论。gemini-code-assist[bot] 自动总结确认变更正确，维护者wuxibin89直接批准。

- 暂无高价值评论线程

风险与影响

- 风险：
  - 核心路径变更风险：修改了 FSDP 引擎在 use_remove_padding=False 且 pad_mode=NO_PADDING 下的 attention mask 构造逻辑，这是训练前向的关键路径。但变更仅替换了 mask 生成方式，不影响其他模式（如 use_remove_padding=True）。
  - 嵌套张量契约：build_attention_mask_from_nested 通过 input_ids.offsets().diff() 推导序列长度，这要求 input_ids 保持 jagged layout。在受影响的路径中 input_ids 始终是

嵌套的，断言 input_ids.is_nested 可提前捕获违反契约的情况。

- 类型安全：max_seq_len 显式转换为 int，避免之前 max() 返回 Python int 而 .item() 可能返回标量张量导致后续操作的类型混淆。
- GPU 测试覆盖不足的风险：目前 GPU 测试仅在单卡 V100 上运行，没有覆盖多 GPU 或分布式场景，但单卡测试已能暴露 mask 的基本正确性问题。

- 影响：
  - 用户影响：修复了使用 use_remove_padding=False + pad_mode=NO_PADDING + strategy=fsdp2 组合的训练用户面临的训练不稳定（超高 KL/clipfrac）问题。变更无需修改配置或 API，行为自动矫正。
  - 系统影响：无，仅限 FSDP 引擎特定路径。
  - 团队影响：新增的辅助函数 build_attention_mask_from_nested 可复用于其他需要从嵌套张量构建 mask 的场景。
  - 风险标记：核心路径变更，依赖嵌套张量契约

关联脉络

- 暂无明显关联 PR