

PR #7055 完整报告

verl-project/verl

[hardware] fix: fix build_npu_attn_mask oom

合并时间: 2026-07-16 14:40

原文链接: <http://prhub.com.cn/verl-project/verl/pull/7055>

执行摘要

- 一句话: 修复 Ascend 上 attention mask 构建 OOM
- 推荐动作: 该 PR 是正确的最小化修复, 建议合并。关注点: 后续可考虑采纳 gemini 建议进一步优化峰值内存, 但当前方案已解决问题, 可先合并。

功能与动机

修复 verl/models/mcore/util.py 中 _build_npu_attn_mask 函数因 float32 中间张量导致的 NPU OOM。PR body 指出 torch.ones 默认 float32, 分配 [seq_len, seq_len] 的 float32 张量 (4 字节 / 元素) 后再 .to(torch.bool) 产生第二个 bool 张量, 峰值内存是实际需要的两倍。

实现拆解

1. 定位 verl/models/mcore/util.py 的 _build_npu_attn_mask 函数第 566 行:
`causal_mask = torch.tril(torch.ones(seq_len, seq_len,
device=original_attention_mask.device)).to(torch.bool)`
2. 将 torch.ones 调用的 dtype 参数设置为 torch.bool, 直接创建布尔类型张量:
`torch.tril(torch.ones(seq_len, seq_len, device=original_attention_mask.device,
dtype=torch.bool))`
3. 移除显式的 .to(torch.bool) 转换, 消除中间 float32 张量的分配。配套变更: 无测试、配置或部署改动。

关键文件:

- verl/models/mcore/util.py (模块 模型; 类别 source; 类型 core-logic): 唯一修改文件, 修复 _build_npu_attn_mask 函数中的 OOM 问题。修改行直接决定因果掩码的构建方式。

关键符号: _build_npu_attn_mask

关键源码片段

verl/models/mcore/util.py

唯一修改文件, 修复 _build_npu_attn_mask 函数中的 OOM 问题。修改行直接决定因果掩码的构建方式。

```
def _build_npu_attn_mask(original_attention_mask: torch.Tensor) -> torch.Tensor:  
    """Build attn_mask for torch_npu.npu_fusion_attention (B1SS / [B, 1, Sq, Skv])"""
```

```
_, seq_len = original_attention_mask.shape
# 原代码：先创建 float32 张量再 .to(torch.bool)，导致两倍峰值内存
# 修改后：直接通过 dtype=torch.bool 创建 bool 张量，峰值内存减半
causal_mask = torch.tril(torch.ones(seq_len, seq_len, device=original_attention_mask.device,
dtype=torch.bool))
attn_mask = original_attention_mask.unsqueeze(-1) & original_attention_mask.unsqueeze(-2)
attn_mask = attn_mask & causal_mask
return (~attn_mask).unsqueeze(1).contiguous()
```

评论区精华

Review 中 `gemini-code-assist[bot]` 建议使用 `torch.arange` 构造网格以减少两次分配（`ones + tril`），进一步减半峰值内存。作者未采纳，PR 被 `wucong25` 直接批准。

- 使用 `torch.arange` 替代 `torch.tril` 进一步优化 (performance): 未采纳，作者未回复，PR 直接被 `wucong25` 批准合并。

风险与影响

- 风险：变更极小（1 行），仅修改 `dtype` 参数，输出语义一致。未引入新依赖或 API 变化。但若未来有代码依赖 `torch.ones` 默认 `float32` 行为（如隐式类型转换），可能受影响（当前场景无此风险）。长序列下 `torch.tril` 仍分配一个 `bool` 张量，若序列极长仍可能 OOM，但已是当前最小侵入修复。
- 影响：直接影响 Ascend NPU 上长序列训练 / 推理，避免 OOM，提升稳定性。不影响 GPU 或其他硬件路径，因为 `_build_npu_attn_mask` 仅在 NPU 场景调用。变更简单，无回退风险。
- 风险标记：缺少测试覆盖，核心路径变更

关联脉络

- 暂无明显关联 PR