

PR #7620 完整报告

verl-project/verl

[model] fix: use the transformers 5 rotary API in the GLM-4V attention patch

合并时间: 2026-09-02 10:06

原文链接: <http://prhub.com.cn/verl-project/verl/pull/7620>

执行摘要

- 一句话: 修复 GLM-4V 注意力层使用已废弃 transformers 4.x 旋转位置编码 API 导致的训练崩溃。
- 推荐动作: 此 PR 值得精读, 尤其是遇到类似依赖库重构后导致 API 不兼容的场景时。它展示了如何精准定位一个因上游破坏性变更而产生的导入错误, 并通过少量、正确的代码修改进行修复。核心设计决策是遵循上游变更, 删除冗余的兼容层代码, 而不是试图恢复旧函数。

功能与动机

GLM-4V 训练在第一个注意力层就会失败, 因为 `glm4v_attn_forward` 导入的 `apply_multimodal_rotary_pos_emb` 在所有被支持的 transformers 5.x 版本中均不存在 (`ImportError`)。此问题由 #5724 (transformers upgrade to 5.3.0) 引入, 升级了配置侧的键名但未同步更新注意力侧的 API 调用。

实现拆解

1. 入口与问题定位: 问题位于 `verl/models/transformers/glm4v.py` 中的 `glm4v_attn_forward` 函数。当使用 `remove_padding` 或 `ulysses_sp_size > 1` 时, 该函数会替换 GLM-4V 模型的注意力前向逻辑, 是训练的核心路径。
2. 核心逻辑修改: 将导入语句从 `apply_multimodal_rotary_pos_emb` 更改为 `apply_rotary_pos_emb`。随后, 删除了为获取旧函数所需的 `mrope_section` 参数而编写的一整段兼容性逻辑 (包括对 `self.rope_scaling` 和 `self.rope_parameter` 的 `getattr` 检查)。这是因为 transformers 5.x 的 `Glm4vTextRotaryEmbedding` 已在生成 `cos/sin` 时处理了 `mrope_section`, 新的 `apply_rotary_pos_emb` 函数无需再接收该参数。最终调用简化为 `apply_rotary_pos_emb(query_states, key_states, cos, sin)`。
3. 测试与配套: 作者最初提交了一个 CPU 级别的回归测试文件 `tests/models/test_glm4v_rotary_on_cpu.py`, 用于验证修复后函数行为正确。然而, 在评审过程中, 评审者认为此测试“CI not needed”, 因此该测试文件被移除。本次变更未引入其他配置、部署或 schema 更改。

关键文件:

- `verl/models/transformers/glm4v.py` (模块 模型层; 类别 source; 类型 data-contract; 符号 `glm4v_attn_forward`): 包含了导致 GLM-4V 训练崩溃的注意力补丁函数 `glm4v_attn_forward`, 是本次修复的核心源码文件。

关键符号: glm4v_attn_forward

关键源码片段

verl/models/transformers/glm4v.py

包含了导致 GLM-4V 训练崩溃的注意力补丁函数 `glm4v_attn_forward`，是本次修复的核心源码文件。

```
# verl/models/transformers/glm4v.py - glm4v_attn_forward 函数片段（修复后）
def glm4v_attn_forward(
    self: "Glm4vTextAttention",
    hidden_states: torch.Tensor,
    attention_mask: Optional[torch.Tensor] = None,
    position_ids: Optional[torch.LongTensor] = None,
    position_embeddings: Optional[tuple[torch.Tensor, torch.Tensor]] = None,
    **kwargs,
) -> tuple[torch.Tensor, None, None]:
    # 关键修复：更新导入为 transformers 5.x 的旋转位置编码函数
    from transformers.models.glm4v.modeling_glm4v import apply_rotary_pos_emb, repeat_kv

    bsz, q_len, _ = hidden_states.size()
    query_states = self.q_proj(hidden_states)
    key_states = self.k_proj(hidden_states)
    value_states = self.v_proj(hidden_states)

    query_states = query_states.view(bsz, q_len, self.num_heads, self.head_dim).transpose(1, 2)
    key_states = key_states.view(bsz, q_len, self.num_key_value_heads, self.head_dim).
    transpose(1, 2)
    value_states = value_states.view(bsz, q_len, self.num_key_value_heads, self.head_dim).
    transpose(1, 2)

    # Because the input can be padded, the absolute sequence length depends on the max
    position id.
    # Glm4vTextRotaryEmbedding already folds mrope_section into cos/sin, so this is the plain
    # application, exactly as Glm4vTextAttention.forward does it.
    cos, sin = position_embeddings
    # 关键修复：调用新的旋转函数，无需 mrope_section 参数
    query_states, key_states = apply_rotary_pos_emb(query_states, key_states, cos, sin)
    key_states = repeat_kv(key_states, self.num_key_value_groups)
    value_states = repeat_kv(value_states, self.num_key_value_groups)
    # ... 后续 Flash Attention 逻辑保持不变 ...
```

评论区精华

评审者 wuxibin89 针对作者提交的回归测试文件 `tests/models/test_glm4v_rotary_on_cpu.py` 提出“CI not needed”。作者 alanhuangyoo 表示同意并删除了该测试文件，同时强调修复本身（更改导入和函数调用）是自立的，因为目标函数在现有依赖下根本不存在，因此该补丁修复是明确且必要的。讨论未涉及核心修改逻辑，仅决定了测试的去留。

- 回归测试文件的必要性 (testing): 作者同意移除该测试文件, 认为修复本身 (更改导入和函数调用) 已足够明确。

风险与影响

- 风险:
 1. 回归风险: 极低。本次变更将调用从不存在的 API (`apply_multimodal_rotary_pos_emb`) 切换为与 transformers 5.x 上游实现一致的、存在的 API (`apply_rotary_pos_emb`), 修复了一个阻塞性错误。
 2. 正确性风险: 低。新调用 `apply_rotary_pos_emb(q, k, cos, sin)` 的参数签名与 transformers 5.x 中 `Glm4vTextAttention.forward` 的调用完全一致, 确保了旋转位置编码的数学操作正确。
 3. 测试覆盖风险: 中低。原作者提交的 CPU 测试被移除, 目前缺乏针对 `glm4v_attn_forward` 函数的直接单元测试。虽然主要逻辑 (注意力计算) 由现有 GPU 测试覆盖, 但旋转编码部分现在缺少特定的回归测试。
- 影响:
 1. 用户影响: 直接修复了 GLM-4V 模型训练在所有 transformers 5.x 版本上的崩溃问题, 影响所有使用 GLM-4V 并启用 `remove_padding` 或 `ulysses_sp_size > 1` 功能的用户。
 2. 系统影响: 恢复了 verl 训练框架对 GLM-4V 模型的支持, 确保了核心训练流程的稳定性。
 3. 团队影响: 清晰地展示了依赖库 API 升级后需同步更新下游代码的维护模式, 为后续维护提供了参考。 - 风险标记: 依赖 API 破坏性变更, 测试覆盖不足

关联脉络

- PR #7650 [ci] chore: upgrade transformers==5.9.0: 本 PR 是解决 transformers 升级 (如 #7650 所示) 后引入的 API 兼容性问题的具体修复。
- PR #7654 [ci] chore: fix ci failure with transformers==5.9.0: 同为解决 transformers 升级后引发的问题, 属于同一波依赖升级的配套修复工作。
- PR #5724 transformers upgrade to 5.3.0: 本 PR 指出的问题最初由 #5724 引入, 该 PR 升级了 transformers 版本但未同步更新 GLM-4V 注意力补丁中的 API 调用。