PR #6675 完整报告

verl-project/verl

[rollout, trainer] fix: handle empty token sequences in _pad_token_ids and metric timing

合并时间：2026-06-14 14:56

原文链接：http://prhub.com.cn/verl-project/verl/pull/6675

执行摘要

- 一句话：修复空 token 序列导致的两个崩溃
- 推荐动作：值得合入。同时建议阅读 review 讨论，了解如何处理 tokenizer pad 和 metrics 中类似的边界情况，这些模式在类似项目中具有参考价值。

功能与动机

在 partial_rollout=False 配置下，vLLM 请求可能被中断导致没有生成任何 token，现有代码无法处理空序列。PR body 提到 'tokenizer.pad() with empty input returns a dict with list values instead of tensors' 和 'empty response batch yields zero token counts, causing ZeroDivisionError'。这两个都是有效的边界情况，需要优雅处理。

实现拆解

1. 修复 _pad_token_ids (agent_loop.py): 在函数开头检查 tokens 是否为空，若为空则使用 tokenizer 的 pad_token_id（若不存在则 fallback 到 0）构造全填充的 tensor，跳过 tokenizer.pad() 调用，避免其返回 list 导致下游 .dim() 崩溃。
2. 修复 compute_timing_metrics (metric_utils.py): 在推导 timing_per_token_ms 时，当 num_tokens_of_section[name] 为 0 时返回 0.0 而不是除以零或返回误导性指标。
3. 添加回归测试:
   - 在 test_agent_loop_extra_fields_schema_on_cpu.py 中新增 _FakeTokenizerCustomPad 类（pad_token_id=42）和 test_agent_loop_pad_token_ids_empty_with_non_zero_pad_id 测试，验证空序列使用正确的 pad_id 且 attention_mask 全零。
   - 在 test_metric_utils_on_cpu.py 中新增 test_compute_timing_metrics_zero_tokens 测试，验证零 token 时 per-token 指标返回 0.0。 这些改动相互独立，分别修复两个模块的边界情况。

关键文件：

- verl/experimental/agent_loop/agent_loop.py（模块 Agent 循环；类别 source；类型 core-logic；符号 _pad_token_ids）：核心修复文件：在 _pad_token_ids 中增加空序列处理，避免 tokenizer.pad 返回 list 导致崩溃
- verl/trainer/ppo/metric_utils.py（模块 训练器；类别 source；类型 core-logic；符号 compute_timing_metrics）：修复 compute_timing_metrics 除零异常，当 response

token 数为 0 时返回 0.0

- tests/experimental/agent_loop/test_agent_loop_extra_fields_schema_on_cpu.py（模块 Agent 测试；类别 test；类型 test-coverage；符号 _FakeTokenizerCustomPad, pad, test_agent_loop_pad_token_ids_empty_with_non_zero_pad_id）：新增回归测试，使用自定义 tokenizer（pad_token_id=42）验证空序列处理
- tests/trainer/ppo/test_metric_utils_on_cpu.py（模块 指标测试；类别 test；类型 test-coverage；符号 test_compute_timing_metrics_zero_tokens）：新增回归测试，验证零 token 时 compute_timing_metrics 返回 0.0

关键符号：_pad_token_ids, compute_timing_metrics

关键源码片段

verl/experimental/agent_loop/agent_loop.py

核心修复文件：在 _pad_token_ids 中增加空序列处理，避免 tokenizer.pad 返回 list 导致崩溃

```
def _pad_token_ids(
    self,
    tokens: list[int],
    *,
    max_length: int,
    padding_side: str,
    return_attention_mask: bool,
) -> dict[str, torch.Tensor]:
    """Right/left pad a flat list of token ids to a ``(1, max_length)`` tensor."""
    # tokenizer.pad() with empty input returns dict with list values
    # instead of tensors, which breaks downstream .dim() calls.
    if not tokens:
        # 使用 tokenizer 的 pad_token_id，若为 None 则 fallback 到 0
        pad_id = self.tokenizer.pad_token_id if self.tokenizer.pad_token_id is not None else 0
        result = {"input_ids": torch.full((1, max_length), pad_id, dtype=torch.long)}
        if return_attention_mask:
            result["attention_mask"] = torch.zeros((1, max_length), dtype=torch.long)
        return result
    self.tokenizer.padding_side = padding_side
    padded = self.tokenizer.pad(
        {"input_ids": tokens},
        padding="max_length",
        max_length=max_length,
        return_tensors="pt",
        return_attention_mask=return_attention_mask,
    )
    if padded["input_ids"].dim() == 1:
        padded["input_ids"] = padded["input_ids"].unsqueeze(0)
        if return_attention_mask:
            padded["attention_mask"] = padded["attention_mask"].unsqueeze(0)
    return padded
```

verl/trainer/ppo/metric_utils.py

修复 compute_timing_metrics 除零异常，当 response token 数为 0 时返回 0.0

```
def compute_timing_metrics(batch: DataProto, timing_raw: dict[str, float]) -> dict[str, Any]:
    # ... 前面的代码计算 num_prompt_tokens, num_response_tokens, num_overall_tokens
    # ... 以及 num_tokens_of_section 字典
    return {
        **{f"timing_s/{name}": value for name, value in timing_raw.items()},
        **{
            f"timing_per_token_ms/{name}": (
                timing_raw[name] * 1000 / num_tokens_of_section[name] if num_tokens_of_
                section[name] > 0 else 0.0
            )
            for name in set(num_tokens_of_section.keys()) & set(timing_raw.keys())
        },
    }
```

评论区精华

讨论集中在两个设计决策上：

- pad_token_id 选择: 最初使用 torch.zeros 填充，但 reviewer 指出可能使用了错误的 token ID（如 Llama-3 的 pad_token_id 不是 0）。作者随后改为使用 tokenizer.pad_token_id，并增加默认值为 0。
- 零 token 指标: 最初使用 max(num_tokens, 1) 避免除零，但 reviewer 指出这会导致在无 token 时报告整个 timing 作为 per-token 值，具有误导性。作者改为在 token 数为 0 时返回 0.0。 此外，reviewer 要求为两个修复添加回归测试，作者均已添加并确认。
- 使用正确的 pad_token_id 而非硬编码 0 (correctness): 改用 self.tokenizer.pad_token_id，若为 None 则 fallback 到 0
- 零 token 时返回 0.0 而非 misleading 值 (correctness): 当 token 数为 0 时返回 0.0
- 添加回归测试覆盖边界情况 (testing): 添加了两个回归测试，覆盖空序列 pad 和零 token 指标

风险与影响

- 风险：风险较低。所有新增代码仅在空 token 序列的边界条件下执行，不影响正常路径。在 _pad_token_ids 中 fallback 到 0 的行为与之前代码默认一致，且 pad_token_id 获取进行了 None 检查，避免 AttributeError。在 compute_timing_metrics 中，当 token 数为 0 时返回 0.0，不会影响其他计算结果。主要风险来自 tokenizer 的 pad_token_id 可能为 None 但代码已处理，以及潜在的其他依赖空序列返回 tensor 格式的代码，但测试覆盖了这些情况。
- 影响：影响范围限于使用 partial_rollout=False 且遇到 vLLM 请求中断的场景。对此类用户，该 PR 修复了训练过程中断的 bug，使训练能够正常完成。对其他用户无影响。团队内合并后可以减少相关 issue 报告。
- 风险标记：边缘路径变更，需测试覆盖

关联脉络

- PR #6718 [fully_async] fix: check audio forwarding contract in agent loop: 同为 agent_loop 模块的修复，关注边界情况处理