PR #7044 完整报告

verl-project/verl

[rollout] fix: handle malformed Qwen3 XML tool calls

合并时间：2026-07-17 17:00

原文链接：http://prhub.com.cn/verl-project/verl/pull/7044

执行摘要

- 一句话：修复 Qwen3 XML 工具解析器对畸形结构的崩溃
- 推荐动作：值得快速合并的 bugfix，设计决策清晰——采用与 vLLM 社区相同的修复方案。建议阅读 _parse_xml_function_call 函数中 .find() 与 .index() 的替换模式，以及 None 守卫的处理方式，可作为类似解析器防御性编程的参考。

功能与动机

Qwen3 模型在 agent 循环中有时会生成残缺的 XML 工具调用（如 <parameter=truncated 缺少 > 分隔符），原有代码使用 str.index(">") 会直接抛出 ValueError: substring not found，并被外层 except Exception 捕获，导致整个工具调用提取结果被丢弃，即使同一调用中仍有合法参数。该问题在 vLLM issue #39771 中被报告，社区已在 vLLM PR #39772 中修复，但 VERL 维护独立的解析器，故需自行修补。

实现拆解

变更涉及两个文件的局部修改：

1. verl/experimental/agent_loop/tool_parser.py：
   - 将 _parse_xml_function_call 的返回类型从 FunctionCall 改为 Optional[FunctionCall]，允许返回 None。
   - 在方法开头添加 tools = tools or []，将 None 归一化为空列表，防止 get_arguments_config 中迭代 None 引发 TypeError。
   - 函数名提取：将 str.index(">") 替换为 str.find(">")，若返回 -1（未找到 >），记录 warning 并返回 None。
   - 参数循环：同样将 str.index(">") 替换为 str.find(">")，若返回 -1，记录 warning 并通过 continue 跳过该畸形参数，保留已解析的有效参数。
   - 在 extract_tool_calls 中，添加一行过滤 None 结果的列表推导：tool_calls = [t for t in tool_calls if t is not None]。
2. tests/experimental/agent_loop/test_qwen3_tool_parser_on_cpu.py：
   - 新增三个测试函数：test_malformed_parameter_is_skipped_but_valid_parameter_is_preserved（验证畸形参数被跳过，合法参数保留）、test_malformed_function_header_returns_none（验证畸形函数头返回 None 且不抛异常）、test_missing_tools_defaults_to_empty_list（验证 tools=None 时能正常解析）。

关键文件：

- verl/experimental/agent_loop/tool_parser.py（模块 工具解析器；类别 source；类型 core-logic；符号 _parse_xml_function_call, extract_tool_calls）：核心修复所在，修改了 _parse_xml_function_call 的返回值类型、添加 None 守卫、替换 .index() 为 .find() 并跳过畸形结构。
- tests/experimental/agent_loop/test_qwen3_tool_parser_on_cpu.py（模块 测试；类别 test；类型 test-coverage；符号 test_malformed_parameter_is_skipped_but_valid_parameter_is_preserved, test_malformed_function_header_returns_none, test_missing_tools_defaults_to_empty_list）：新增三个单元测试，覆盖畸形参数保留、畸形函数头跳过和 tools=None 场景，确保修复的正确性和 API 兼容性。

关键符号：_parse_xml_function_call, extract_tool_calls

关键源码片段

verl/experimental/agent_loop/tool_parser.py

核心修复所在，修改了 _parse_xml_function_call 的返回值类型、添加 None 守卫、替换 .index() 为 .find() 并跳过畸形结构。

```
def _parse_xml_function_call(
    self, function_call_str: str, tools: Optional[list[OpenAIFunctionToolSchema]]
) -> Optional[FunctionCall]:
    # 将 None 归一化为空列表，防止后续迭代 tools 时 TypeError
    tools = tools or []

    # ... 内部函数定义 ...

    # 尝试提取函数名。使用 find 而非 index，这样找不到 > 时不会崩溃
    end_index = function_call_str.find(">")
    if end_index == -1:
        # 函数头不完整，无法恢复，跳过并记录警告
        logger.warning(f"Skipping malformed function call without '>' separator: {function_call_
        str!r}")
        return None

    function_name = function_call_str[:end_index]
    param_config = get_arguments_config(function_name)
    parameters = function_call_str[end_index + 1 :]
    param_dict = {}
    for match in self.tool_call_parameter_regex.findall(parameters):
        match_text = match[0] if match[0] else match[1]
        # 同样使用 find，若找不到 > 则跳过该畸形参数，但保留已解析的有效参数
        idx = match_text.find(">")
        if idx == -1:
            logger.warning(
                f"Skipping malformed parameter without '>' separator in tool call for function "
                f"'{function_name}': {match_text!r}"
```

```
            )
            continue
        param_name = match_text[:idx]
        param_value = str(match_text[idx + 1 :])
        # ... 值转换逻辑 ...
        param_dict[param_name] = convert_param_value(param_value, param_name, param_
        config, function_name)

    return FunctionCall(name=function_name, arguments=json.dumps(param_dict))
```

评论区精华

review 过程中，gemini-code-assist[bot] 指出一个高优先级问题：若 tools 参数为 None（extract_tool_calls 的默认值），在 get_arguments_config 中迭代 tools 会引发 TypeError: 'NoneType' object is not iterable。建议在 _parse_xml_function_call 入口处添加 tools = tools or []。该建议被采纳，且已在最终代码中实现。除此之外，PR 作者 wuxibin89 已批准合并。无未解决疑虑。

- tools=None 导致 TypeError (correctness): 已采纳，在 _parse_xml_function_call 第一行添加 tools = tools or []。

风险与影响

- 风险：变更范围极小，仅涉及工具解析器的两处字符串查找逻辑和一处 None 守卫，无性能或安全风险。由于模型输出天然不可预测，跳过畸形参数并保留合法参数是合理的降级策略，不会引入回归。测试文件仅新增对 CPU 的单元测试，不影响 GPU 或 NPU 环境。
- 影响：影响范围仅限于使用 Qwen3 或 Qwen3.5 模型且启用 agent_loop 工具调用的场景。修复后，模型输出的畸形 XML 工具调用不会导致整个提取失败，合法参数仍能被成功解析并传递给工具执行。对不使用工具调用的用户无影响。对系统整体稳定性有正面提升。
- 风险标记：核心路径变更

关联脉络

- PR #6434 [rollout] fix: Reasoning block stripping for Qwen3 XML tool parser.: 同样是针对 Qwen3 XML 工具解析器的修复，涉及推理块剥离，但与此 PR 修复的畸形 XML 解析问题不同。
- PR #7038 [rollout, vllm] fix: stop the policy from sampling vision placeholder tokens: 同样属于 rollout 模块的 bugfix，涉及模型输出 token 的过滤，但关注点不同（视觉占位符 token）。