PR #42026 完整报告

vllm-project/vllm

[Bugfix] Preserve leading/trailing whitespace in GLM non-streaming tool parser

合并时间：2026-05-09 12:49

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/42026

执行摘要

- 一句话：修复 GLM 非流式工具解析器去除首尾空格
- 推荐动作：建议精读。这是一个典型的细小但重要的 bugfix，展示了如何通过审慎的条件判断解决语义边界问题。其改动模式（从 " 统一 strip" 到 " 根据类型条件化 strip"）值得学习。此外，review 中关于 _is_string_type 的讨论指出了进一步改进方向。

功能与动机

PR body 指出，当模型生成包含首尾空格的字符串参数（如缩进代码）时，解析器会错误地移除这些空格，导致解析结果不正确。例如 <arg_value> indented code </arg_value> 应解析为 " indented code " 而非 "indented code"。

实现拆解

1. 修改核心解析逻辑：在 vllm/tool_parsers/glm4_moe_tool_parser.py 的 *extract_tool_calls* 方法中，将原来对所有参数值统一 value.strip() 并调用 _deserialize 的逻辑，改为先通过 _is_string_type 判断是否为字符串类型：如果是字符串类型，则直接使用原始 value（不 strip）；如果不是，则 strip 后调用 _deserialize。
2. 为 GLM4 解析器添加测试：在 tests/tool_parsers/test_glm4_moe_tool_parser.py 中添加 test_whitespace_preserved_in_arg_values 测试函数，使用包含首尾空格的字符串参数，验证解析结果保留原始空格。
3. 为 GLM47 解析器添加测试：在 tests/tool_parsers/test_glm47_moe_tool_parser.py 中添加同名测试函数，同样验证首尾空格保留。

关键文件：

- vllm/tool_parsers/glm4_moe_tool_parser.py（模块 工具解析器；类别 source；类型 *core-logic*）：核心变更文件，修改了 *extract_tool_calls* 方法中的参数值处理逻辑，区分字符串与非字符串类型的 strip 行为。
- tests/tool_parsers/test_glm4_moe_tool_parser.py（模块 测试；类别 test；类型 *test-coverage*；符号 test_whitespace_preserved_in_arg_values）：为 GLM4 解析器添加了首尾空格保留的测试用例，确保修复可靠。
- tests/tool_parsers/test_glm47_moe_tool_parser.py（模块 测试；类别 test；类型 *test-coverage*；符号 test_whitespace_preserved_in_arg_values）：为 GLM47 解析器添加类似测试用例，确保双回归覆盖。

关键符号：extract_tool_calls, test_whitespace_preserved_in_arg_values

关键源码片段

vllm/tool_parsers/glm4_moe_tool_parser.py

核心变更文件，修改了 extract_tool_calls 方法中的参数值处理逻辑，区分字符串与非字符串类型的 strip 行为。

```
# vllm/tool_parsers/glm4_moe_tool_parser.py ( 关键片段 )

def extract_tool_calls(
    self,
    model_output: str,
    request: ChatCompletionRequest,
) -> ExtractedToolCallInformation:
    matched_tool_calls = self.func_call_regex.findall(model_output)
    logger.debug("model_output: %s", model_output)
    try:
        tool_calls: list[ToolCall] = []
        for match in matched_tool_calls:
            tc_detail = self.func_detail_regex.search(match)
            if not tc_detail:
                logger.warning(
                    "Failed to parse tool call details from: %s",
                    match,
                )
                continue
            tc_name = tc_detail.group(1).strip()
            tc_args = tc_detail.group(2)
            pairs = self.func_arg_regex.findall(tc_args) if tc_args else []
            arg_dct: dict[str, Any] = {}
            for key, value in pairs:
                arg_key = key.strip()
                # 关键变更：先检查是否为字符串类型
                # 若是字符串类型，保留原始 value（含首尾空格）
                # 否则 strip 后反序列化
                if self._is_string_type(tc_name, arg_key, self.tools):
                    arg_val = value
                else:
                    # 非字符串类型：strip 掉空格后尝试反序列化
                    arg_val = self._deserialize(value.strip())
                logger.debug("arg_key = %s, arg_val = %s", arg_key, arg_val)
                arg_dct[arg_key] = arg_val
            tool_calls.append(
                ToolCall(
                    type="function",
                    function=FunctionCall(
                        name=tc_name,
                        arguments=json.dumps(arg_dct, ensure_ascii=False),
```

```
                ),
            )
        )
        # ... 后续异常处理和返回逻辑保持不变
```

tests/tool_parsers/test_glm4_moe_tool_parser.py

为 GLM4 解析器添加了首尾空格保留的测试用例，确保修复可靠。

```
# tests/tool_parsers/test_glm4_moe_tool_parser.py ( 新增测试 )

def test_whitespace_preserved_in_arg_values(glm4_moe_tokenizer):
    """Test that string arguments preserve leading and trailing whitespace."""
    tools = [
        ChatCompletionToolsParam(
            function=FunctionDefinition(
                name="apply_diff",
                parameters={
                    "type": "object",
                    "properties": {
                        "s": {"type": "string"}, # 定义字符串类型参数
                    },
                    "required": ["s"],
                },
            ),
        ),
    ]
    parser = Glm4MoeModelToolParser(glm4_moe_tokenizer, tools=tools)
    request = ChatCompletionRequest(model=MODEL, messages=[], tools=tools)

    # 模拟模型输出，包含首尾空格的字符串参数
    model_output = """<tool_call>apply_diff
<arg_key>s</arg_key>
<arg_value>    indented code    </arg_value>
</tool_call>"""

    extracted_tool_calls = parser.extract_tool_calls(model_output, request=request)
    args = json.loads(extracted_tool_calls.tool_calls[0].function.arguments)

    # 预期值应保留首尾空格
    assert args["s"] == "    indented code    "
```

评论区精华

reviewer bbrowning 指出，_is_string_type 逻辑目前未处理可空字符串类型（如 ["string", "null"]），但认为此为超出当前 PR 范围的问题，建议后续处理。已有关联 PR #40197 涉及类似修复，但该 PR 包含其他变更需进一步审查。

- 可空字符串类型的处理 (design): 该问题超出当前 PR 范围，建议后续另开 PR 处理。已有关联 PR #40197 涉及此问题，但需拆分后单独合并。

风险与影响

- 风险：风险较低。变更仅影响 GLM4 和 GLM47 的非流式工具解析器的字符串参数处理，且 review 确认该行为与流式路径一致。但需注意 _is_string_type 对可空字符串类型的处理不足，可能导致此类参数仍被 strip，属于已知局限。
- 影响：影响范围：仅限使用 GLM4 和 GLM47 模型且使用非流式工具调用的场景。修复后，字符串类型参数将正确保留首尾空格，避免了代码编辑等场景的解析错误。对非字符串参数行为无影响。影响程度：中等，修复了一个明确的错误。
- 风险标记：已知局限：可空字符串类型未处理

关联脉络

- PR #40197 [Bugfix] Fix nullable string type detection in tool parsers: reviewer 提及该 PR 包含对可空字符串类型检测的修复，但包含其他变更，需拆分为独立 PR。此 PR 与 42026 在修复方向上互补。