

PR #53444 完整报告

vllm-project/vllm

[Bugfix] Handle bare and malformed tool call openers in Gemma4 parser

合并时间: 2026-09-15 10:52

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/53444>

执行摘要

- 一句话: 修复 Gemma4 解析器静默丢弃裸 tool_call
- 推荐动作: 值得精读。该 PR 展示了 FSM 解析器应对模型非规范输出的一种最小干预模式: 单个终端 + 单条转移即可修复整类静默丢失问题, 且依赖 lexer 最长匹配优先级保证零回归。建议关注两点: 一是 TOOL_PREAMBLE -> TOOL_NAME 转移与 #53657 逃逸舱口的协同设计, 二是测试对 "畸形 + 规范混合序列" 的覆盖方式, 这种场景最容易暴露 FSM 状态泄漏。

功能与动机

关联 Issue #53431 报告: [gemma-4-26b-a4b-it](#) 会输出两种 tool call opener 形式——文档化的 `<|tool_call>call:name{...}<|tool_call|>` 和裸形式 `<|tool_call>:name{...}<|tool_call|>`。前者可被解析, 后者被 FSM 静默丢弃, streaming 与非 streaming 均复现; 由于 `TOOL_PREAMBLE` 状态没有 `content_events` 条目, 整个 span 既不产生 tool call 也不产生 content, turn 直接丢失且无任何错误提示。Issue 讨论中 @chriskosys 进一步补充: 无花括号或使用圆括号的畸形调用会让 `function.name` 吸收其后所有内容, 包括关闭 token 和后续规范调用。

实现拆解

1. 新增 `COLON` 终端: 在 `vllm/parser/gemma4.py` 的 `gemma4_config()` 的 `terminals` 字典中新增 `"COLON": ":"` 字面量, 与已有的 `CALL_PREFIX: "call:"`、`OPEN_BRACE`、`OPEN_PAREN` 并列, 供 FSM 在 `TOOL_PREAMBLE` 状态识别裸 opener 的冒号。
2. 新增状态转移: 添加 `(ParserState.TOOL_PREAMBLE, "COLON") -> TOOL_NAME` 转移 (无事件输出)。这样 `<|tool_call>:get_weather{...}` 能像文档化格式一样进入 `TOOL_NAME` 并走原有 `OPEN_BRACE` 转移进入 `TOOL_ARGS`, 工具调用不再被静默丢弃。
3. 依赖 lexer 最长字面量优先: FSM 词法层按字面量长度优先匹配, `"call:"` (5 字符) 仍优先于 `":"` (1 字符), 因此文档化格式 `call:name{...}` 行为完全不变; `COLON` 只在 `TOOL_PREAMBLE` 状态下定义转移, 普通文本和参数值中的冒号不参与该状态转移, 不受影响。
4. 与逃逸舱口协同: `(TOOL_NAME, "TOOL_END") -> CONTENT` 转移已在先前的 #53657 中合入 main, 本 PR rebase 后与之协同: 若模型从未输出 { (无花括号畸形调用), FSM 停在 `TOOL_NAME` 时遇到 `TOOL_END` 即返回 `CONTENT` 并发出 `TOOL_CALL_END`, 畸形调用被自身边界截断, 后续内容和工具调用恢复正常解析, 不会再把关闭 token 和后续调

用吸进 `function.name`。

5. 测试配套: `tests/tool_parsers/test_gemma4_tool_parser.py` 新增

`TestBareAndMalformedOpeners` 类共 9 个测试, 覆盖裸 opener 非流式解析、文档化格式回归、多参数、畸形无花括号边界、畸形 + 规范连续调用、裸畸形 + 规范混合、三个混合调用、内容中冒号不被破坏、空参数花括号。现有 54 个测试全部通过, 无回归。

关键文件:

- `vllm/parser/gemma4.py` (模块 解析器; 类别 `source`; 类型 `core-logic`; 符号 `gemma4_config`) : FSM 配置的源码主路径: 新增 COLON 终端与 (TOOL_PREAMBLE, COLON) 转移, 是修复裸 opener 静默丢弃的核心改动。
- `tests/tool_parsers/test_gemma4_tool_parser.py` (模块 工具解析; 类别 `test`; 类型 `test-coverage`; 符号 `TestBareAndMalformedOpeners`, `test_bare_opener_non_streaming`, `test_documented_form_still_works`, `test_bare_opener_multiple_args`) : 新增 `TestBareAndMalformedOpeners` 9 个回归测试, 覆盖裸 opener、畸形调用边界、混合序列与内容冒号场景, 是本次修复的验证主体。

关键符号: `gemma4_config`, `TestBareAndMalformedOpeners`,
`test_bare_opener_non_streaming`, `test_malformed_then_well_formed`,
`test_three_mixed_calls`

关键源码片段

`vllm/parser/gemma4.py`

FSM 配置的源码主路径: 新增 COLON 终端与 (TOOL_PREAMBLE, COLON) 转移, 是修复裸 opener 静默丢弃的核心改动。

```
@functools.cache
def gemma4_config() -> ParserEngineConfig:
    """构造 Gemma4 工具调用解析器的 FSM 配置 (terminals + transitions) 。”
    return ParserEngineConfig(
        name="gemma4",
        initial_state=ParserState.CONTENT,
        terminals={
            "THINK_START": CHANNEL_START,
            "THINK_END": CHANNEL_END,
            "TOOL_START": TOOL_CALL_START,
            "TOOL_END": TOOL_CALL_END,
            "CALL_PREFIX": "call:",
            # 本次新增: `<tool_call>:name{...}` 裸 opener 直接匹配冒号字面量。
            # lexer 按字面量长度优先匹配, "call:" (5 字符) 先于 ":" (1 字符) 被消费,
            # 因此文档化格式行为不变; 普通文本中的冒号不参与下方转移, 同样不受影响。
            "COLON": ":",
            "OPEN_BRACE": "{",
            "OPEN_PAREN": "(",
        },
        transitions={
            # 文档化格式的既有转移: `<tool_call>call:name{...}` 走 CALL_PREFIX。
```

```

(ParserState.TOOL_PREAMBLE, "CALL_PREFIX"): Transition(
    ParserState.TOOL_NAME, ()
),
# 本次新增的裸 opener 转移: TOOL_PREAMBLE 遇冒号直接进入 TOOL_NAME,
# 与 CALL_PREFIX 一样不产生事件, 后续由 OPEN_BRACE / OPEN_PAREN
# 继续转入 TOOL_ARGS 解析参数。
(ParserState.TOOL_PREAMBLE, "COLON"): Transition(
    ParserState.TOOL_NAME, ()
),
(ParserState.TOOL_NAME, "OPEN_BRACE"): Transition(
    ParserState.TOOL_ARGS, ()
),
(ParserState.TOOL_NAME, "OPEN_PAREN"): Transition(
    ParserState.TOOL_ARGS, ()
),
# 逃逸舱口 (由 PR #53657 合入 main, 本 PR rebase 后与之协同):
# 若模型从未输出 "{", TOOL_END 把 FSM 拉回 CONTENT 并发出
# TOOL_CALL_END, 畸形调用被自身边界截断, 后续内容和工具调用
# 恢复正常解析, 不会再被吸进 function.name。
(ParserState.TOOL_NAME, "TOOL_END"): Transition(
    ParserState.CONTENT, (EventType.TOOL_CALL_END,)
),
(ParserState.TOOL_ARGS, "TOOL_END"): Transition(
    ParserState.CONTENT, (EventType.TOOL_CALL_END,)
),
},
content_events={
    # TOOL_PREAMBLE 没有 content_events 条目: 修复前 FSM 卡在此状态时,
    # 整个 `

```

tests/tool_parsers/test_gemma4_tool_parser.py

新增 TestBareAndMalformedOpeners 9 个回归测试, 覆盖裸 opener、畸形调用边界、混合序列与内容冒号场景, 是本次修复的验证主体。

```

class TestBareAndMalformedOpeners:
    """裸 opener 与畸形调用边界回归测试, 对应 issue #53431。"""

    def test_bare_opener_non_streaming(self, parser, mock_request):
        # 裸形式不带 "call" 前缀, 修复前 FSM 卡在 TOOL_PREAMBLE,
        # 整段调用被静默丢弃; 修复后应正常提取名称与参数。
        model_output = (
            '<lttool_call>:get_weather{location:<|">London<|">}<tool_call>'
        )
        result = parser.extract_tool_calls(model_output, mock_request)

```

```

assert result.tools_called is True
assert len(result.tool_calls) == 1
assert result.tool_calls[0].function.name == "get_weather"
args = json.loads(result.tool_calls[0].function.arguments)
assert args == {"location": "London"}

def test_malformed_then_well_formed(self, parser, mock_request):
    # 无花括号的畸形调用后紧跟规范调用：修复前关闭 token 和后续
    # 调用会被吸进 function.name；修复后靠 TOOL_END 逃逸舱口截断，
    # 第二个调用需要被正常恢复。
    model_output = (
        "<ltool_call>call:bad_func no brace<tool_calll>"
        '<ltool_call>call:get_weather{location:<l">London<l">}'
        "<tool_calll>"
    )
    result = parser.extract_tool_calls(model_output, mock_request)

    assert result.tools_called is True
    assert len(result.tool_calls) == 2
    assert result.tool_calls[0].function.name == "bad_func no brace"
    assert result.tool_calls[1].function.name == "get_weather"
    assert "London" in result.tool_calls[1].function.arguments

```

评论区精华

- 补测试的要求（已解决）：审核人 chaunceyjiang 在 Review 中留言 "Could you add some test code?", 作者随即新增 TestBareAndMalformedOpeners 9 个测试，并列覆盖场景（裸 opener、文档化格式回归、无花括号边界、连续调用恢复、内容冒号等）。
- PR 范围收敛（已解决）：作者在评论区说明 rebase 后发现 TOOL_NAME, TOOL_END 逃逸舱口已由 #53657 合入 main，本 PR 相应收敛为只新增 COLON 终端与对应转移，避免重复实现。
- pre-commit 合规（已解决）：mergify bot 两次提示 pre-commit 检查失败，作者运行格式化并提交 0d7e929 后 CI 通过。
- 最终批准：chaunceyjiang 在补充测试后给出 APPROVED ("LGTM")。
- 要求补充回归测试 (testing)：作者补充 9 个回归测试（裸 opener、文档化格式回归、畸形边界、混合序列、内容冒号、空参数），现有 54 个测试也全部通过。
- PR 范围在 rebase 后收敛 (design)：PR 收敛为最小改动，避免与 #53657 重复实现；两者共同构成完整修复。
- pre-commit 格式检查 (style)：作者提交格式化 commit 0d7e929 后通过，CI 全绿。

风险与影响

- 风险：
 - COLON 是新增的全局字面量终端，理论上参与 FSM 各状态的词法匹配；但转移仅在 TOOL_PREAMBLE 状态定义，且 lexer 最长匹配优先保证 "call:" 不被拆解，风险受控。

测试 `test_colon_in_content_not_broken` 验证了普通文本冒号不受影响。

- 修复效果依赖 #53657 的逃逸舱口：若该改动被回退，或本改动被 cherry-pick 到不含 #53657 的旧分支，无花括号畸形调用仍可能污染后续内容，需要注意合并顺序。
- 新测试集中在非流式 `extract_tool_calls` 路径；流式路径依赖同一 FSM 配置，现有流式测试（如 `test_streaming_trailing_bare_bool_not_duplicated`）全部通过，但裸 opener 的流式场景没有直接覆盖。
- `_gemma4_arg_converter` 对) 结尾的处理逻辑已有 `OPEN_PAREN` 转移配合，本次改动未触碰该逻辑，圆括号格式行为不变。
- 影响：
 - 用户：使用 `gemma-4-26b-a4b-it` 并开启 `--enable-auto-tool-choice --tool-call-parser gemma4` 的用户，此前会静默丢失裸 opener 工具调用（streaming 与非 streaming 均复现），修复后可正常提取；畸形调用不再污染后续内容和工具调用。
 - 系统：改动仅限 `gemma4_config()` 的 FSM 配置，`mistral` 等其他 parser 的 `CALL_PREFIX` 逻辑不受影响。
 - 团队 / 维护：源码 +8 行、测试 +108 行，改动面小；9 个新测试为后续 Gemma4 输出格式变化提供回归护栏，回滚成本低。
 - 风险标记：解析器 FSM 核心路径变更，依赖 lexer 最长匹配优先级，协同依赖 #53657 逃逸舱口，流式裸 opener 未直接覆盖

关联脉络

- PR #53657 [Bugfix] Add TOOL_NAME escape hatch for malformed Gemma4 tool calls: PR 评论中明确提及：该 PR 合入了 `(TOOL_NAME, TOOL_END) -> CONTENT` 逃逸舱口，本 PR rebase 后与之协同，两者共同构成 Gemma4 裸 / 畸形 opener 的完整修复（标题系根据评论内容重建，材料中未给出原标题）。