

PR #55954 完整报告

vllm-project/vllm

[Bugfix] Parse DSML tool calls when the model omits the tool_calls wrapper

合并时间: 2026-09-09 08:38

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/55954>

执行摘要

- 一句话: 修复 DeepSeek V4/V3.2 工具解析器在模型省略启动包装器时泄漏原始 DSML 文本的问题。
- 推荐动作: 此 PR 值得精读, 因为它修复了一个影响用户生产环境的 bug, 并展示了如何通过扩展状态机来优雅地处理模型输出的不确定性。其测试策略非常全面, 涵盖了多种边界情况, 为类似问题的解决提供了良好范例。

功能与动机

关联 Issue #48931 报告, DeepSeek-V4-Flash 在长上下文 (~95k+ tokens) 下会间歇性地省略 `< | DSML | tool_calls>` 启动标记, 但依然输出格式正确的 `< | DSML | invoke ...>` 块。由于原有的解析逻辑仅依赖启动标记进行工具调用检测, 导致整个工具调用块作为纯文本内容返回给客户端, 引发客户端挂起。本 PR 旨在修复此问题, 并统一 V3.2/V4 解析器的行为。

实现拆解

- 核心逻辑修复 (V4 & V3.2 状态机): 在 `vllm/parser/deepseek_v4.py` 和 `vllm/parser/deepseek_v32.py` 的 `ParserEngineConfig` 配置中, 为 `transitions` 字典新增了从 `ParserState.CONTENT` 到 `ParserState.TOOL_NAME` 的转换, 其触发条件为 `INVOKE_PREFIX`。这使得解析器能够直接从普通文本状态识别并进入工具调用解析, 即使缺少启动包装器。
- 行为变更: 丢弃工具块后文本: 将 `TOOL_ARGS` 遇到 `TOOL_END` 时的转换目标从 `CONTENT` 改为 `TOOL_BETWEEN`, 并在 `TOOL_BETWEEN` 遇到 `TOOL_END` 时也停留在 `TOOL_BETWEEN` 状态。这样, 工具调用块之后的任何文本都不会被解析为内容, 从而确保 V4 和 V3.2 在有无结束包装器时行为一致。
- 新增转换以支持连续块: 在 `TOOL_BETWEEN` 状态下新增对 `TOOL_START` 的转换, 以支持一个孤立工具块后跟一个包装器包裹的工具块的场景。
- 全面的测试覆盖: 在 `tests/parser/engine/test_deepseek_v4.py` 中新增了 `TestMissingToolCallsWrapper` 测试类, 覆盖了非流式、流式、尾随文本丢弃、并行孤立调用、推理模式下的孤立调用等多种边界情况。在 `tests/parser/engine/test_deepseek_v32.py` 中新增了 `TestMissingFunctionCallsWrapper` 测试类。同时更新了 `tests/tool_parsers/test_deepseekv4_tool_parser.py` 中的测试用例以反映新的预期行为。

关键文件:

- `vllm/parser/deepseek_v4.py` (模块 解析器; 类别 source; 类型 core-logic) : V4 解析器的核心状态机配置, 包含修复启动包装器缺失和尾随文本丢弃的关键状态转换逻辑。
- `vllm/parser/deepseek_v32.py` (模块 解析器; 类别 source; 类型 core-logic) : V3.2 解析器的状态机配置, 包含与 V4 类似的核心修复逻辑。
- `tests/parser/engine/test_deepseek_v4.py` (模块 解析器; 类别 test; 类型 test-coverage ; 符号 `TestMissingToolCallsWrapper`, `test_non_streaming`, `test_trailing_content_dropped_consistently`, `test_second_wrapped_block_after_orphan`) : V4 解析器核心测试文件, 新增了全面的回归测试以覆盖启动包装器缺失的各种边缘情况。

关键符号: `deepseek_v4_config`, `deepseek_v32_config`

关键源码片段

`vllm/parser/deepseek_v4.py`

V4 解析器的核心状态机配置, 包含修复启动包装器缺失和尾随文本丢弃的关键状态转换逻辑。

以下是 `vllm/parser/deepseek_v4.py` 中 `deepseek_v4_config()` 函数内 `transitions` 字典的关键变更。

变更前: 工具调用检测严格依赖 `TOOL_START` 包装器。

变更后: 新增从 `CONTENT` 状态直接识别 `INVOKE_PREFIX` 的转换, 使工具调用检测锚定在 `invoke` 标记上。

```
transitions={
    # ... 其他转换 ...
    (ParserState.CONTENT, "TOOL_START"): Transition(
        ParserState.TOOL_PREAMBLE,
        (),
    ),
    # 新增: 从普通文本状态直接识别工具调用开始标记, 处理模型省略启动包装器的情况。
    (ParserState.CONTENT, "INVOKE_PREFIX"): Transition(
        ParserState.TOOL_NAME,
        (EventType.TOOL_CALL_START,),
    ),
    # ... 其他转换 ...
    # 修改: 工具参数结束后进入 TOOL_BETWEEN 而非 CONTENT, 为丢弃尾随文本做准备。
    (ParserState.TOOL_ARGS, "TOOL_END"): Transition(
        ParserState.TOOL_BETWEEN,
        (EventType.TOOL_CALL_END,),
    ),
    # 修改: 在 TOOL_BETWEEN 状态遇到 TOOL_END 时停留, 丢弃块后文本。
    (ParserState.TOOL_BETWEEN, "TOOL_END"): Transition(
        ParserState.TOOL_BETWEEN,
        (),
    ),
    # 新增: 支持一个孤立工具块后跟一个包装器包裹的工具块的场景。
    (ParserState.TOOL_BETWEEN, "TOOL_START"): Transition(
        ParserState.TOOL_PREAMBLE,
        (),
    ),
}
```

```
}
```

vllm/parser/deepseek_v32.py

V3.2 解析器的状态机配置，包含与 V4 类似的核心修复逻辑。

以下是 vllm/parser/deepseek_v32.py 中 deepseek_v32_config() 函数内 transitions 字典的关键变更。

变更逻辑与 V4 解析器保持一致。

```
transitions={
    # ... 其他转换 ...
    (ParserState.CONTENT, "TOOL_START"): Transition(
        ParserState.TOOL_PREAMBLE,
        (),
    ),
    # 新增：从普通文本状态直接识别工具调用开始标记。
    (ParserState.CONTENT, "INVOKE_PREFIX"): Transition(
        ParserState.TOOL_NAME,
        (EventType.TOOL_CALL_START,),
    ),
    # ... 其他转换 ...
    # 修改：工具参数结束后进入 TOOL_BETWEEN 而非 CONTENT。
    (ParserState.TOOL_ARGS, "TOOL_END"): Transition(
        ParserState.TOOL_BETWEEN,
        (EventType.TOOL_CALL_END,),
    ),
    # 修改：在 TOOL_BETWEEN 状态遇到 TOOL_END 时停留，丢弃块后文本。
    # 新增：支持一个孤立工具块后跟一个包装器包裹的工具块的场景。
    (ParserState.TOOL_BETWEEN, "TOOL_START"): Transition(
        ParserState.TOOL_PREAMBLE,
        (),
    ),
}
```

评论区精华

核心讨论围绕对不完整工具调用的处理展开。评审者 yewentao256 提出了一个边界情况：...< | DSML | invoke name="terminal">not a real call（缺少结束标记）。作者 sfeng33 解释，在这种情况下，terminal 将被解析为工具调用，参数为空对象，而 Documentation: 作为内容返回。评审者最终接受了这种行为，认为即使没有 </ | DSML | invoke> 结束标记，工具调用仍然可以被使用，这与官方 DSML 实现相匹配。

- 不完整工具调用（缺少结束标记）的处理策略 (correctness): 评审者接受该行为，认为即使没有结束标记，工具调用仍可使用，这与官方实现一致。

风险与影响

- 风险：风险主要在于行为变更可能影响现有依赖工具块后文本的客户端。本 PR 将工具块后的文本从内容中丢弃，这是一个有意的行为变更，但可能对某些客户端产生兼容性影响。此外，新逻辑依赖状态机转换的正确性，需要确保在所有边缘情况（如流式传输中的标记拆分、

推理模式下的孤立调用)下都能正常工作。当前的测试覆盖较为全面,降低了回归风险。

- 影响: 此变更直接影响使用 DeepSeek V4 和 V3.2 模型进行工具调用的用户。它修复了一个严重的 bug, 即工具调用被错误地作为文本返回, 导致客户端挂起。对于正确场景(有完整启动包装器的工具调用), 行为保持不变。对于新的边缘情况(缺少启动包装器), 现在能够正确解析。丢弃工具块后文本的行为变更可能对某些边缘用例产生影响, 但符合预期语义。
- 风险标记: 行为变更, 状态机逻辑复杂

关联脉络

- PR #41240 [Bugfix] Handle string="true" attribute in DSML tool calls: 处理 DSML 工具调用中的 string="true" 属性, 是同一解析器的早期修复。
- PR #41801 [Bugfix] Repair malformed tool_calls wrapper in DSML output: 修复格式错误的 tool_calls 包装器, 是同一解析器的早期修复。