PR #22595 完整报告

sgl-project/sglang

fix: normalize tool message content for GLM5.1 chat template

合并时间：2026-04-16 16:48

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/22595

执行摘要

- 一句话：归一化工具消息内容从数组格式到字符串，修复 GLM5.1 等聊天模板问题。
- 推荐动作：建议工程师精读此 PR，重点关注 normalize_tool_content 函数的设计决策，如如何通过检查 type == "text" 来区分文本部分和结构化列表，以及单元测试的全面覆盖，这对于处理 API 兼容性问题和消息格式归一化有借鉴意义。

功能与动机

根据 OpenAI API 规范，工具消息的 content 字段支持字符串或内容部分数组两种格式。许多客户端（如 Claude Code / Cursor）使用数组格式，但模型聊天模板（如 GLM-5 和 GLM-5.1 的 Jinja 模板）只正确处理字符串格式，导致工具结果无法正确渲染，模型持续生成工具调用循环。SGLang 作为 OpenAI 兼容 API 层应接受所有有效格式，并归一化为字符串以确保模板工作，避免逐个修补上游模板。

实现拆解

1. 新增归一化函数：在 python/sglang/srt/entrypoints/openai/serving_chat.py 中新增 normalize_tool_content 函数，检查消息角色是否为 "tool" 且内容是否为列表，如果是，则验证所有部分是否为 OpenAI 文本部分（类型为 "text" 的字典或字符串），若是则拼接为字符串（使用空格分隔），否则保留原列表。
2. 集成到消息处理流程：在 _apply_jinja_template 方法中调用 normalize_tool_content，在消息处理流程中归一化工具消息内容，确保传递给聊天模板的内容为字符串格式。
3. 添加单元测试：在 test/registered/openai_server/basic/test_serving_chat.py 中添加 TestNormalizeToolContent 单元测试类，覆盖多种场景：文本部分扁平化、多部分拼接、非文本部分保留、字符串内容不变、空列表返回空字符串、非工具角色不变、混合字符串和字典部分，确保逻辑正确性和边缘情况处理。

关键文件：

- python/sglang/srt/entrypoints/openai/serving_chat.py（模块 OpenAI 服务；类别 source；类型 core-logic；符号 normalize_tool_content）：核心逻辑文件，新增归一化函数并在消息处理中调用，确保工具消息内容从数组格式转换为字符串。
- test/registered/openai_server/basic/test_serving_chat.py（模块 测试套件；类别 test；类型 test-coverage；符号 TestNormalizeToolContent, test_openai_text_parts_flattened, test_multiple_text_parts_joined,

test_non_text_part_list_preserved）：测试配套文件，添加单元测试验证归一化逻辑的正确性和覆盖各种场景。

关键符号：normalize_tool_content

关键源码片段

python/sglang/srt/entrypoints/openai/serving_chat.py

核心逻辑文件，新增归一化函数并在消息处理中调用，确保工具消息内容从数组格式转换为字符串。

```
def normalize_tool_content(role: str, content):
    """Normalize tool message content from OpenAI array format to plain string.

    OpenAI clients may send tool content as a list of content parts
    (e.g. [{"type":"text","text":"..."}]) but most chat templates expect
    a plain string for tool messages. Only flatten when ALL items are
    pure OpenAI text parts; preserve lists containing non-text-type items
    that some templates intentionally iterate over.
    """
    if role != "tool" or not isinstance(content, list):
        return content # 非工具角色或非列表内容直接返回，无需处理
    parts = content
    is_openai_text_parts = all(
        (isinstance(p, dict) and p.get("type") == "text") or isinstance(p, str)
        for p in parts
    ) # 检查所有部分是否为OpenAI文本部分（字典类型为"text"或字符串）
    if is_openai_text_parts:
        text_parts = [p.get("text", "") if isinstance(p, dict) else p for p in parts]
        return " ".join(text_parts) # 拼接为字符串，使用空格分隔以保持一致性
    return content # 否则保留原列表，如包含非文本部分的结构化列表
```

test/registered/openai_server/basic/test_serving_chat.py

测试配套文件，添加单元测试验证归一化逻辑的正确性和覆盖各种场景。

```
class TestNormalizeToolContent(unittest.TestCase):
    """Unit tests for normalize_tool_content()."""

    def test_openai_text_parts_flattened(self):
        # 测试单个OpenAI文本部分数组被扁平化为字符串
        result = normalize_tool_content("tool", [{"type": "text", "text": "10525"}])
        self.assertEqual(result, "10525")

    def test_multiple_text_parts_joined(self):
        # 测试多个文本部分拼接为字符串
        result = normalize_tool_content(
            "tool",
            [{"type": "text", "text": "hello"}, {"type": "text", "text": "world"}],
        )
```

```
        self.assertEqual(result, "hello world")

    def test_non_text_part_list_preserved(self):
        # 测试非文本部分列表被保留，如结构化工具语义字段
        content = [{"name": "func", "output": "result"}]
        result = normalize_tool_content("tool", content)
        self.assertIs(result, content) # 确保原列表引用不变

    # 其他测试方法类似，覆盖字符串内容不变、空列表、非工具角色、混合部分等场景
```

评论区精华

Review 中主要讨论了三个点：注释准确性、分隔符一致性和单元测试覆盖。JustinTong0323 指出原注释误描述过滤逻辑（基于 name/output 字段），建议修正为 'preserve lists containing non-text-type items'，同时建议分隔符使用空格以与 jinja_template_utils.py 保持一致；作者在第二个 commit 中响应了这些建议，调整了注释和分隔符。ShangmingCai 询问是否适用于 dpsk v32，作者回复 dpsk v32 不使用聊天模板机制，因此本修复不影响其特殊处理。所有讨论点均已解决，无未解决疑虑。

- 注释准确性和分隔符一致性 (design): 作者修正了注释以准确描述基于 type == "text" 的过滤逻辑，并将分隔符从换行符改为空格，确保代码一致性。
- 单元测试覆盖 (testing): 作者添加了 TestNormalizeToolContent 类，包含七个测试方法，全面覆盖了文本部分扁平化、非文本部分保留等场景。
- dpsk v32 适用性 (correctness): 确认本修复仅针对使用聊天模板的模型，dpsk v32 由于其特殊处理不受影响，无需额外调整。

风险与影响

- 风险：技术风险较低：回归风险小，因为只针对工具消息内容进行归一化，且保留了非文本部分列表，不影响其他角色或格式；性能影响可忽略，新增逻辑简单，仅增加少量条件检查和字符串操作；无安全风险；兼容性方面，改善了 OpenAI API 兼容性，但需注意 dpsk v32 等不使用模板的场景不受影响，设计决策已明确区分。
- 影响：对用户而言，使用 OpenAI 兼容客户端的用户将看到工具调用结果正确传递，避免无限循环问题，提升工具使用体验；对系统，修复了关键功能缺陷，确保模型能接收工具结果，提高稳定性和兼容性；对团队，通过集中修复避免了逐个修补模型模板的维护成本，简化了 API 层处理逻辑。
- 风险标记：格式兼容性修复，已添加测试覆盖

关联脉络

- 暂无明显关联 PR