PR #27451 完整报告

sgl-project/sglang

Classify malformed-multimodal rejects as invalid_request

合并时间：2026-06-07 01:19

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27451

执行摘要

- 一句话：结构化多模态错误响应与客户端断开处理
- 推荐动作：值得精读，展示了如何通过异常分层与请求状态检查来提升 API 的健壮性和客户端体验。

功能与动机

多模态请求中如 base64 格式错误等校验失败时，原先返回的是裸错误消息，不利于客户端结构化处理；此外流式请求中客户端断开连接会错误地触发 400 响应。PR 旨在统一响应格式并区分客户端错误与服务器错误。

实现拆解

1. 异常区分：在 base_processor.py 的 _load_single_item 方法中，先捕获 ValueError（如无效 base64），重新抛出带上下文信息的 ValueError；原有 Exception 则抛出 RuntimeError 用于内部错误。
2. 流式错误结构化：在 http_server.py 的 generate_request 流式分支中，捕获 ValueError 后先检查客户端是否断开，若是则记录并返回；否则返回结构化 JSON 包含 type: invalid_request_error、code: 400、retryable: false。
3. 数据截断：在 base_processor.py 的两个异常路径中，将 data 截断为 100 字符后再放入错误消息，防止大 base64 导致日志膨胀。
4. 非流式路径保持不变：非流式分支仍使用 _create_error_response，存在与流式路径的不一致。

关键文件：

- python/sglang/srt/multimodal/processors/base_processor.py（模块 多模态处理器；类别 source；类型 core-logic；符号 _load_single_item）：核心多模态加载逻辑，区分 ValueError 与 RuntimeError，并截断错误消息中的 data。
- python/sglang/srt/entrypoints/http_server.py（模块 HTTP 服务；类别 source；类型 core-logic；符号 stream_results, generate_request）：流式请求错误处理，检测客户端断开并返回结构化 400 错误。

关键符号：_load_single_item, stream_results, generate_request

关键源码片段

python/sglang/srt/multimodal/processors/base_processor.py

核心多模态加载逻辑，区分 ValueError 与 RuntimeError，并截断错误消息中的 data。

```
# python/sglang/srt/multimodal/processors/base_processor.py (head)
# _load_single_item 方法中异常处理部分的变更

        try:
            if modality == Modality.IMAGE:
                img, _ = load_image(data, cls.gpu_image_decode)
                if discard_alpha_channel and not isinstance(img, torch.Tensor) and img.mode !=
                "RGB":
                    img = img.convert("RGB")
                return img
            elif modality == Modality.VIDEO:
                return load_video(data, frame_count_limit)
            elif modality == Modality.AUDIO:
                return load_audio(data, audio_sample_rate)

        except ValueError as e:
            # 捕获 ValueError（如无效 base64），作为客户端错误抛出，确保调用方返回 4xx
            data_str = str(data)
            if len(data_str) > 100:
                data_str = data_str[:100] + "..."
            raise ValueError(f"Error while loading data {data_str}: {e}") from e
        except Exception as e:
            # 其他异常（加载失败等）作为 RuntimeError，标记为 5xx 内部错误
            data_str = str(data)
            if len(data_str) > 100:
                data_str = data_str[:100] + "..."
            raise RuntimeError(f"Error while loading data {data_str}: {e}") from e
```

python/sglang/srt/entrypoints/http_server.py

流式请求错误处理，检测客户端断开并返回结构化 400 错误。

```
# python/sglang/srt/entrypoints/http_server.py (head)
# generate_request 中流式分支的变更

        async def stream_results() -> AsyncIterator[bytes]:
            try:
                async for out in _global_state.tokenizer_manager.generate_request(
                    obj, request
                ):
                    yield b"data: " + dumps_json(out) + b"

"
            except ValueError as e:
                # 客户端断开连接在此处也可能被捕获（流式取消），不是服务器错误或坏输入
                if request is not None and await request.is_disconnected():
                    logger.info(f"[http_server] Client disconnected: {e}")
```

```
                return # 不发送任何错误事件
            # 这才是真正的客户端输入错误，返回结构化 400
            out = {
                "error": {
                    "message": str(e),
                    "type": "invalid_request_error",
                    "code": 400,
                    "retryable": False,
                }
            }
            logger.error(f"[http_server] Error: {e}")
            yield b"data: " + dumps_json(out) + b"

"
        yield b"data: [DONE]

"
```

评论区精华

Review 中 gemini-code-assist[bot] 提出两个改进点：

1. 截断 base_processor.py 中异常消息的 data 字符串，以防大 base64 导致日志和内存问题（高优先级）。此建议已在第二 commit 中实现。
2. 统一流式与非流式错误响应格式，确保 API 一致性（中等优先级）。此建议未被采纳。

- 错误消息中 data 字符串截断 (performance): 已采纳，第二 commit 实现了 100 字符截断。
- 流式与非流式错误格式一致性 (design): 未采纳，此版本保持非流式路径不变。

风险与影响

- 风险：低风险。变更集中于错误处理路径，不影响正常请求流程。但非流式路径未同步更新，可能导致同一 API 流式与非流式错误格式不一致，对客户端解析造成轻微混淆。
- 影响：影响所有使用多模态功能（图像、视频、音频）的请求，特别是流式生成场景。客户端现在能收到结构化的 400 错误，方便自动化处理；同时客户端断开不再产生错误日志。
- 风险标记：错误处理路径变更，缺少测试覆盖

关联脉络

- 暂无明显关联 PR