PR #42478 完整报告

vllm-project/vllm

[Bugfix] Fix Qwen3-ASR transcription streaming postprocessing

合并时间：2026-07-09 15:33

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/42478

执行摘要

- 一句话：修复 Qwen3-ASR 流式转录输出前缀泄漏
- 推荐动作：该 PR 已合并，值得精读以理解 vLLM 中音频流式后处理的设计模式。重点关注 Qwen3ASRStreamingPostProcessor.process_delta 的三态机逻辑和增量输出策略，以及 serving.py 中如何集成后处理器并调整分隔符。对于需要实现自定义 ASR 模型后处理的开发者，该 PR 提供了清晰的扩展点。

功能与动机

关联 Issue #35767 报告 Qwen3-ASR 实时端点输出退化，其中流式 SSE delta 包含 language ...<asr_text> 框架文本，客户端需额外清理。PR body 明确指出："..where RequestOutputKind.DELTA chunks were sent without model-specific post-processing"，此 PR 旨在修复该问题。

实现拆解

1. 定义流式后处理器接口：在 vllm/model_executor/models/interfaces.py 中新增 StreamingTranscriptionPostProcessor 基类和 process_delta 默认实现，并在 SupportsTranscription 协议中添加 get_streaming_post_processor_cls 类方法，默认返回无操作处理器，保证所有转录模型向后兼容。
2. 实现 Qwen3-ASR 后处理器：在 vllm/model_executor/models/qwen3_asr.py 中新增 Qwen3ASRStreamingPostProcessor 类，持有 raw_text、emitted_text 和 is_structured_output 状态。process_delta 方法首先累积输入，然后判断是否为结构化前缀（以 language 开头、长度上限 50、不含换行），若正在缓冲且未找到 <asr_text> 标记则返回空字符串；一旦确定结构化输出，调用 _post_process_qwen3_asr_output 剥离前缀，仅返回新增部分。非结构化文本直接透传。
3. 集成到流式服务：在 vllm/entrypoints/speech_to_text/base/serving.py 中，SpeechToTextBaseServing.__init__ 缓存 streaming_post_processor_cls；_speech_to_text_stream_generator 在每个音频块的流开始前实例化后处理器，对每个 RequestOutput 调用 process_delta，将返回结果作为 delta content；仅当清理后的 delta 非空时插入块间分隔符；若 delta 为空且未完成则跳过发送（仍计数 token）。
4. 补充测试：在 tests/entrypoints/speech_to_text/transcription/test_transcription_inter_chunk_spacing.py 中新增 6 个单元测试，覆盖纯文本透传、前缀跨多帧缓冲、独立处理器状态、不完整前缀完成时回退、超长前缀停止缓冲、含换行前缀停止缓冲等场景。

关键文件：

- vllm/model_executor/models/qwen3_asr.py（模块 模型实现；类别 source；类型 data-contract；符号 _post_process_qwen3_asr_output, Qwen3ASRStreamingPostProcessor, init, process_delta）：核心实现：新增 Qwen3ASRStreamingPostProcessor 类，包含 __init__ 和 process_delta 方法；新增辅助函数 _post_process_qwen3_asr_output；修改 post_process_output 以复用新函数；新增 get_streaming_post_processor_cls 类方法。
- vllm/model_executor/models/interfaces.py（模块 接口定义；类别 source；类型 data-contract；符号 StreamingTranscriptionPostProcessor, process_delta, get_streaming_post_processor_cls）：定义 StreamingTranscriptionPostProcessor 基类和 get_streaming_post_processor_cls 钩子，为所有转录模型提供默认行为。
- vllm/entrypoints/speech_to_text/base/serving.py（模块 服务层；类别 source；类型 core-logic）：集成流式后处理器：在 __init__ 中缓存 post-processor 类，在 _speech_to_text_stream_generator 中实例化并调用 process_delta，调整分隔符插入逻辑。
- tests/entrypoints/speech_to_text/transcription/test_transcription_inter_chunk_spacing.py（模块 测试；类别 test；类型 test-coverage；符号 test_qwen3_asr_stream_processor_passes_plain_text_without_prefix, test_qwen3_asr_stream_processor_buffers_prefix_with_leading_space, test_qwen3_asr_stream_processor_keeps_independent_state, test_qwen3_asr_stream_processor_emits_finished_incomplete_prefix）：新增 6 个单元测试覆盖流式后处理器的各种场景，确保稳定性。

关键符号：_post_process_qwen3_asr_output, Qwen3ASRStreamingPostProcessor.init, Qwen3ASRStreamingPostProcessor.process_delta, StreamingTranscriptionPostProcessor.process_delta, SupportsTranscription.get_streaming_post_processor_cls, Qwen3ASRForConditionalGeneration.get_streaming_post_processor_cls, SpeechToTextBaseServing.init, SpeechToTextBaseServing._speech_to_text_stream_generator

关键源码片段

vllm/model_executor/models/qwen3_asr.py

核心实现：新增 Qwen3ASRStreamingPostProcessor 类，包含 __init__ 和 process_delta 方法；新增辅助函数 _post_process_qwen3_asr_output；修改 post_process_output 以复用新函数；新增 get_streaming_post_processor_cls 类方法。

```
# 流式后处理器：累积输入、判断结构化前缀、增量输出
class Qwen3ASRStreamingPostProcessor(StreamingTranscriptionPostProcessor):
    def __init__(self) -> None:
        self.raw_text = "" # 累积原始文本
        self.emitted_text = "" # 已发出的处理后文本
        self.is_structured_output: bool | None = None # None: 未知，True: 结构化，False:
```

```
        非结构化

    def process_delta(self, text_delta: str, finished: bool) -> str:
        self.raw_text += text_delta

        if self.is_structured_output is None:
            # 判断是否可能是结构化输出（以 "language " 开头）
            text_without_leading_space = self.raw_text.lstrip()
            maybe_structured_output = (
                text_without_leading_space == ""
                or _LANGUAGE_PREFIX.startswith(text_without_leading_space)
                or (
                    text_without_leading_space.startswith(_LANGUAGE_PREFIX)
                    and len(text_without_leading_space) < _MAX_STREAMING_PREFIX_CHARS #
                    避免无限缓冲
                    and "\n" not in text_without_leading_space # 换行视为文本结束
                )
            )
            if maybe_structured_output and _ASR_TEXT_TAG not in self.raw_text:
                if not finished:
                    return "" # 继续缓冲
                self.is_structured_output = False
            else:
                self.is_structured_output = _ASR_TEXT_TAG in self.raw_text

        # 根据 is_structured_output 选择处理方式
        processed_text = (
            _post_process_qwen3_asr_output(self.raw_text) # 剥离前缀，只保留 <asr_text> 之后
            if self.is_structured_output
            else self.raw_text
        )
        # 只返回新增部分，避免重复
        new_text = processed_text[len(self.emitted_text):]
        self.emitted_text = processed_text
        return new_text
```

vllm/model_executor/models/interfaces.py

定义 StreamingTranscriptionPostProcessor 基类和 get_streaming_post_processor_cls 钩子，为所有转录模型提供默认行为。

```
class StreamingTranscriptionPostProcessor:
    """Stateful streaming post-processor for transcription deltas."""
    def process_delta(self, text_delta: str, finished: bool) -> str:
        # 默认实现：直接返回输入
        return text_delta

class SupportsTranscription(Protocol):
    # ... 其他方法 ...
```

```
    @classmethod
    def get_streaming_post_processor_cls(
        cls,
    ) -> type[StreamingTranscriptionPostProcessor]:
        # 默认返回无操作处理器，子类可覆盖
        return StreamingTranscriptionPostProcessor
```

评论区精华

1. 闭包 vs 类设计：Reviewer NickLucche 指出初始实现使用 nonlocal 闭包管理状态是 "weird"，建议改为显式状态类或 StreamingPostProcessor 抽象。作者采纳建议，重构为 Qwen3ASRStreamingPostProcessor 类。NickLucche 最终 approved。
2. 文本被吞风险：gemini-code-assist[bot] 指出当 <asr_text> 出现后 processed_text 可能短于 emitted_text，导致 new_text 为空而丢失部分转录。最终实现通过 is_structured_output 三态机处理：在结构化判定前普通文本不发出（返回空），标记一旦确认则从累积中提取转录，利用 emitted_text 增量输出，避免了丢失。
3. 缓冲过度导致延迟：同一 bot 指出若无 <asr_text> 标记（如幻觉或用户实际说 "language"）会无限缓冲。最终实现增加了 _MAX_STREAMING_PREFIX_CHARS = 50 和 "\n" in text_without_leading_space 检查，超出长度或含换行即视为非结构化文本直接透传。

- 闭包 vs 类的设计选择 (design): 作者重构为 Qwen3ASRStreamingPostProcessor 类，获得 NickLucche 批准。
- 文本被吞的潜在问题 (correctness): 最终实现通过 is_structured_output 三态机确保在结构化判定前普通文本不发出，切换时不会丢失已发送文本。空 delta 在服务中被跳过，不影响后续输出。
- 缓冲过度导致流式延迟 (performance): 添加 _MAX_STREAMING_PREFIX_CHARS = 50 和 \n 检查，超出限制或含换行则视为非结构化文本立即透传。

风险与影响

- 风险：
  1. 非结构化文本误判：如果普通文本恰好以 language 开头且长度小于 50 且不含换行，后处理器会误判为结构化前缀而缓冲，直到出现换行或超长或 finishe。这可能导致短暂延迟（最多 <50 chars），但概率较低且最终会回退。
  2. 状态独立于音频块：每个音频块创建新的后处理器实例，状态不跨块共享。若需跨块上下文（如 Issue #35767 提到的 realtime 场景）则不在本 PR 范围内，但当前 REST 流式设计如此，无风险。
  3. 默认回退为空操作：其他转录模型未覆盖 get_streaming_post_processor_cls 时使用默认无操作处理器，行为不变，无回归。
  4. 文本变短边缘情况：如前文所述，当 is_structured_output 从 None 切换为 True 时，processed_text 可能短于 emitted_text，导致一次空 delta。但因为在判定结构化前普通文本已缓冲未发出，实际上不会丢失已发送文本。客户端收到空 delta 会忽略，但 token 计数仍在，后续 delta 正常。该行为可接受。 - 影响：用户影响：Qwen3-ASR 的流式转录响应现在只返回清理后的纯文本，客户端可直接拼接 SSE delta，无需额外解析

前缀。系统影响：新增 get_streaming_post_processor_cls 钩子，所有 SupportsTranscription 实现类自动获得默认实现，无侵入。团队影响：未来新增 ASR模型若需流式后处理，只需覆盖该钩子并返回自定义处理器类。 - 风险标记：核心路径变更，边界状态处理，缺少回退机制（已处理），模型定制化

关联脉络

- PR #35767 [Enhancement]: Qwen3-ASR realtime endpoint produces degraded output — stateless segments, no cross-segment context, raw format leaks: 此 PR 的直接关联 Issue，报告 Qwen3-ASR 实时端点输出退化，其中 raw format leak 是修复重点。
- PR #35894 [Bugfix] Fix Qwen3-ASR realtime streaming: PR body 提及的相关 PR，目标为 Qwen3-ASR 实时流式路径，与本 PR 的 REST 流式路径互补。
- PR #36018 [Feature] Add response_prefix parameter for audio transcription/translation endpoints: PR body 提及的相关 PR，添加 response_prefix 参数，与本 PR 的模型侧后处理不同。