PR #32611 完整报告

sgl-project/sglang

Fix transcription & audio-understanding for ASR/audio/speech models

合并时间：2026-08-19 18:44

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/32611

执行摘要

- 一句话：为 Qwen2-Audio / GLM-ASR / Granite Speech 补齐真实 ASR 转录
- 推荐动作：值得精读。该 PR 展示了在多模态服务框架中为“不同架构的同类任务”做适配器隔离的典型做法：适配器注册 + 契约方法（build_sampling_params / build_verbose_response）+ 空提示兜底 + 时长预算缩放。尤其建议关注 resolve_adapter 的子串匹配风险、以及 Qwen2-Audio 固定窗口截断这类“上游约束下放”到 warning 的设计取舍。

功能与动机

PR body 明确指出：此前这三个 decoder-only 语音模型在转录端点会“falling back to the Whisper adapter”，而 Whisper 专用参数（_detect_language、强制语言 / 任务前缀）这些模型并不支持；同时 /v1/audio/transcriptions 发送的 text="" 不携带音频占位符，导致编码器特征无处填充、提示构造失败。因此需要为每个模型实现专属适配器与提示兜底逻辑，才能提供真实可用的 ASR 服务。

实现拆解

实现按以下 5 个步骤展开：

1. 新增转录适配器注册：新建 python/sglang/srt/entrypoints/openai/transcription_adapters/glmasr.py、granite_speech.py、qwen2_audio.py，分别以GlmAsr、GraniteSpeech、Qwen2Audio 为注册键并实现 supports_language_detection=False；每个适配器实现 build_sampling_params（temperature + 按音频时长缩放且带 448 token 下限的 max_new_tokens）与 build_verbose_response（language=None、空 segments）；GLM-ASR 额外实现 postprocess_text 剥离模型输出中的 assistant 前缀与引号，降低 WER。transcription_adapters/init.py 同步完成导出与注册装配。
2. 空文本提示兜底：针对 /v1/audio/transcriptions 的 text="" 场景，在 qwen_audio.py 与多模态 processors/glmasr.py 中新增 _build_transcription_prompt()，通过 apply_chat_template 渲染带音频占位符的对话模板；transformers_auto.py 新增 audio 分支与 _build_audio_prompt()，并用 audio_token_index 扩展音频 token 查找，保证 Granite Speech 路径也能插入 <|audio|> 占位符。
3. 处理器兼容性修复：base_processor.py 将 GraniteSpeechProcessor 加入 audio= 关键字白名单，因为它使用的是新式 audio= 参数而非已废弃的 audios=。

4. 模型侧音频特征对齐：重写 python/sglang/srt/models/glmasr.py 中 GlmAsrForConditionalGeneration.get_audio_feature，按 30s 窗口重组编码器输出、依据 input_features_mask 经 conv 子采样与 merge_factor 降采样保留有效嵌入，使嵌入数量与音频占位符 token 数一致；Qwen2-Audio processor 强制 truncation=True 将超过 30s 的音频裁剪到固定 3000 mel 窗口，并新增 _warn_if_audio_exceeds_window 警告日志。
5. 测试配套：新增 test/registered/unit/entrypoints/openai/test_transcription_adapters.py，注册到 CPU CI（suite=base-a-test-cpu，est_time=4），覆盖适配器解析与 Whisper 回退、语言检测关闭、max_new_tokens 下限与时长缩放边界，以及 verbose-response 契约。

关键文件：

- python/sglang/srt/entrypoints/openai/transcription_adapters/glmasr.py（模块 转录适配器；类别 source；类型 core-logic；符号 GlmAsrAdapter, build_sampling_params, postprocess_text, build_verbose_response）：新增 GLM-ASR 专用转录适配器，核心包含按时长缩放 max_new_tokens、postprocess_text 前缀剥离与 verbose 响应契约，是 PR 主干逻辑之一。
- python/sglang/srt/entrypoints/openai/transcription_adapters/granite_speech.py（模块 转录适配器；类别 source；类型 core-logic；符号 GraniteSpeechAdapter, build_sampling_params, build_verbose_response）：新增 Granite Speech 专用转录适配器，实现相同的时长缩放采样参数与 verbose 契约，是三类模型支持的关键组成。
- python/sglang/srt/entrypoints/openai/transcription_adapters/qwen2_audio.py（模块 转录适配器；类别 source；类型 core-logic；符号 Qwen2AudioAdapter, build_sampling_params, build_verbose_response）：新增 Qwen2-Audio 专用转录适配器，与 GLM-ASR / Granite 共用时长缩放策略，是 /v1/audio/transcriptions 对 Qwen2-Audio 生效的直接保障。
- python/sglang/srt/entrypoints/openai/transcription_adapters/__init__.py（模块 适配器注册；类别 source；类型 dependency-wiring）：适配器注册装配点，新增导入使 GlmAsr / GraniteSpeech / Qwen2Audio 三个适配器进入 resolve_adapter() 的分发范围。
- test/registered/unit/entrypoints/openai/test_transcription_adapters.py（模块 适配器测试；类别 test；类型 test-coverage；符号 TestTranscriptionAdapterResolution, test_resolves_expected_adapter_per_architecture, test_unknown_architecture_falls_back_to_whisper, TestSpeechLMAdapterContract）：PR 的唯一测试配套，以 CPU CI 覆盖适配器解析、Whisper 回退、采样参数 floor 与时长缩放边界、verbose 契约，防止注册键子串碰撞与协议回归。
- python/sglang/srt/multimodal/processors/qwen_audio.py（模块 音频处理器；类别 source；类型 core-logic；符号 _build_transcription_prompt, _warn_if_audio_exceeds_window）：为 Qwen2-Audio 处理器增加空文本转录提示兜底与 30s 截断警告，是该模型在转录端点可用的关键配套修改。
- python/sglang/srt/multimodal/processors/transformers_auto.py（模块 通用处理器；类别 source；类型 core-logic；符号 _build_audio_prompt, process_mm_data_async）：为通用 HF 自动处理器补齐音频数据分支与 _build_audio_prompt 占位符兜底，并扩展

audio_token_index 查找到 audio_token_id，是 Granite Speech 路径能够工作的基础。

- python/sglang/srt/models/glmasr.py（模块 语音模型；类别 source；类型 data-contract；符号 get_audio_feature）：重写 GlmAsrForConditionalGeneration.get_audio_feature，按 30s 窗口重组编码器输出并依据 input_features_mask 降采样保留有效嵌入，是嵌入数与占位符 token 对齐的关键。
- python/sglang/srt/multimodal/processors/glmasr.py（模块 音频处理器；类别 source；类型 core-logic；符号 _build_transcription_prompt）：为 GLM-ASR 处理器补齐 _build_transcription_prompt，与 Qwen2-Audio 类似处理空文本转录请求，是 GLM-ASR 转录可用的提示侧保障。
- python/sglang/srt/multimodal/processors/base_processor.py（模块 处理器基类；类别 source；类型 core-logic）：将 GraniteSpeechProcessor 加入 audio= 关键字白名单，避免其 HF 处理器因收到废弃 audios= 参数而失败。

关键符号：GlmAsrAdapter.build_sampling_params, GlmAsrAdapter.postprocess_text, GlmAsrAdapter.build_verbose_response, GraniteSpeechAdapter.build_sampling_params, GraniteSpeechAdapter.build_verbose_response, Qwen2AudioAdapter.build_sampling_params, Qwen2AudioAdapter.build_verbose_response, Qwen2AudioMultimodalProcessor._build_transcription_prompt, Qwen2AudioMultimodalProcessor._warn_if_audio_exceeds_window, GlmAsrProcessor._build_transcription_prompt, AutoMultimodalProcessor._build_audio_prompt, AutoMultimodalProcessor.process_mm_data_async, GlmAsrForConditionalGeneration.get_audio_feature

关键源码片段

python/sglang/srt/entrypoints/openai/transcription_adapters/glmasr.py

新增 GLM-ASR 专用转录适配器，核心包含按时长缩放 max_new_tokens、postprocess_text 前缀剥离与 verbose 响应契约，是 PR 主干逻辑之一。

```
# 转录适配器：GLM-ASR
# 注册键 GlmAsr 会被 resolve_adapter() 按架构字符串子串匹配命中。
# 语音大约 2~3 词 / 秒，而 tokenizer 每个词会拆出多于 1 个 token，
# 因此 15 token/s 的预算既留出充足余量，又限制了失控生成。
_MAX_NEW_TOKENS_PER_SECOND = 15
# 生成长度下限取 448，保证 ~30s 的短音频也能完整转录。
_DEFAULT_MAX_NEW_TOKENS = 448


@register_transcription_adapter("GlmAsr")
class GlmAsrAdapter(TranscriptionAdapter):
    # GLM-ASR 是 decoder-only 语音模型，处理器插入
    # <|begin_of_audio|>...<|end_of_audio|> 占位符后自由生成文本，
    # 没有 Whisper 风格的强制语言 / 任务前缀，所以关闭语言检测。
    _ASSISTANT_PREFIXES = (
        "The spoken content of the audio is",
```

```
        "The transcription of the audio is",
        "The content of the input audio is",
    )

    def build_sampling_params(self, request: TranscriptionRequest) -> dict:
        # /v1/audio/transcriptions 请求侧没有长度字段，只能由适配器控制：
        # 短音频用下限，长音频按时长缩放，避免转录被静默截断。
        duration_s = request.audio_duration_s or 0.0
        return {
            "temperature": request.temperature,
            "max_new_tokens": max(
                _DEFAULT_MAX_NEW_TOKENS,
                int(duration_s * _MAX_NEW_TOKENS_PER_SECOND),
            ),
        }

    def postprocess_text(self, text: str) -> str:
        # 镜像 HF GlmAsrProcessor.decode(strip_prefix=True)，
        # 剥离模型在转写前后包装的 assistant 前缀与引号，降低 WER。
        stripped = text.strip()
        for prefix in self._ASSISTANT_PREFIXES:
            if stripped.startswith(prefix):
                stripped = stripped[len(prefix):].strip()
                break
        if stripped.endswith("."):
            stripped = stripped[:-1].strip()
        if (
            len(stripped) >= 2
            and stripped[0] == stripped[-1]
            and stripped[0] in {"'", '"'}
        ):
            stripped = stripped[1:-1].strip()
        return stripped

    def build_verbose_response(self, request, text, ret, tokenizer, usage):
        # GLM-ASR 内部自行推断语言，前端不声称支持语言检测，
        # 因此 language 置 None，不返回时间戳 segments。
        return TranscriptionVerboseResponse(
            language=None,
            duration=round(request.audio_duration_s, 2),
            text=text,
            segments=[],
            usage=usage,
        )
```

python/sglang/srt/entrypoints/openai/transcription_adapters/granite_speech.py

新增 Granite Speech 专用转录适配器，实现相同的时长缩放采样参数与 verbose 契约，是三类模型支持的关键组成。

```
# Granite Speech 是 English-only 的 decoder-only 语音模型，
# 音频编码器特征合并进 <|audio|> 占位符后自由生成转写文本。
# 它同样没有 Whisper 风格的强制语言 / 任务前缀，因此关闭语言检测。
@register_transcription_adapter("GraniteSpeech")
class GraniteSpeechAdapter(TranscriptionAdapter):
    def build_sampling_params(self, request: TranscriptionRequest) -> dict:
        # 转录端点没有请求侧长度字段，只能由适配器按音频时长控制
        # max_new_tokens：短音频取下限 448，长音频按时长线性缩放。
        duration_s = request.audio_duration_s or 0.0
        return {
            "temperature": request.temperature,
            "max_new_tokens": max(
                _DEFAULT_MAX_NEW_TOKENS,
                int(duration_s * _MAX_NEW_TOKENS_PER_SECOND),
            ),
        }

    def build_verbose_response(self, request, text, ret, tokenizer, usage):
        # language 置 None：模型会从音频中自行推断语言，
        # 适配器不声明它没有真正兑现的语言字段。
        return TranscriptionVerboseResponse(
            language=None,
            duration=round(request.audio_duration_s, 2),
            text=text,
            segments=[],
            usage=usage,
        )
```

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

为 Qwen2-Audio 处理器增加空文本转录提示兜底与 30s 截断警告，是该模型在转录端点可用的关键配套修改。

```
# Qwen2-Audio 的音频塔要求恰好 3000 mel 帧（固定 30s 窗口），
# BaseMultimodalProcessor 默认传 truncation=False（分块编码器需要），
# 超过 30s 的片段会超出窗口导致音频塔报错，因此这里强制截断。
self.audio_config = {**self.audio_config, "truncation": True}

# 转录专用对话模板：chat template 只匹配裸 audio 键，
# 严格指令可避免模型输出 "The content of this audio is:" 前缀，
# 从而控制 WER 膨胀。
_TRANSCRIPTION_CONVERSATION = [
    {
        "role": "user",
        "content": [
            {"type": "audio", "audio": ""},
            {
```

```
                "type": "text",
                "text": (
                    "Transcribe the audio. Output only the exact transcription, "
                    "with no preamble, prefix, commentary, or quotation marks."
                ),
            },
        ],
    }
]

def _build_transcription_prompt(self, input_text) -> str:
    # /v1/audio/transcriptions 端点发送空文本（input_ids 也为空），
    # 没有音频占位符可供编码器特征填充；这里渲染带一个 audio span
    # 的聊天提示作为兜底，否则走调用方传入的原文。
    if isinstance(input_text, list):
        input_text = (
            self._processor.tokenizer.decode(input_text) if input_text else ""
        )
    if input_text and input_text.strip():
        return input_text
    return self._processor.apply_chat_template(
        self._TRANSCRIPTION_CONVERSATION,
        add_generation_prompt=True,
        tokenize=False,
    )

def _warn_if_audio_exceeds_window(self, audios) -> None:
    # Qwen2-Audio 编码器是单个固定 30s 窗口，超长输入只转录前 30s，
    # 这里按采样率与 chunk_length 换算最大样本数并给出显式警告。
    feature_extractor = self._processor.feature_extractor
    max_samples = int(
        feature_extractor.sampling_rate * feature_extractor.chunk_length
    )
    for audio in audios:
        if isinstance(audio, np.ndarray) and audio.shape[-1] > max_samples:
            logger.warning(
                "Qwen2-Audio input is %.1fs but the encoder window is %ds; "
                "only the first %ds will be transcribed (audio truncated).",
                audio.shape[-1] / feature_extractor.sampling_rate,
                feature_extractor.chunk_length,
                feature_extractor.chunk_length,
            )
```

评论区精华

核心讨论集中在三处：

1. GLM-ASR 输出前缀：mickqian 建议“Could we override postprocess_text to mirror HF's strip_prefix=True? Otherwise GLM-ASR may return 'The spoken content of the

audio is "..."' instead of the raw transcript, which hurts WER”，SKRohit 回应已按 qwen3_asr 的方式处理，最终提交里加入了 _ASSISTANT_PREFIXES 剥离。

2. Qwen2-Audio 30 秒截断：mickqian 指出强制 truncation=True 后超过 30 秒的音频只会转录前 30 秒，SKRohit 说明该限制来自 transformers 的 Qwen2AudioProcessor 固定窗口，并补充截断警告日志作为缓解。
3. verbose 响应语言字段：mickqian 质疑适配器在 supports_language_detection=False 时回显用户语言是在“声称从未被兑现的语言”，建议返回 None 或真正传递 hint；最终统一改为 language=None，并由测试固定该契约。

- Qwen2-Audio 强制 truncation 导致超过 30s 音频只转录前 30s (correctness): 保留强制截断以规避音频塔报错，同时在 _warn_if_audio_exceeds_window 中输出显式警告；这是上游处理器约束下的折中方案。
- GLM-ASR 输出带 assistant 前缀，需要 postprocess_text 剥离 (correctness): 新增 _ASSISTANT_PREFIXES 前缀剥离与引号清理，最终提交 'Address review: GLM prefix strip' 落实该修改。
- verbose 响应回显请求语言字段是否真实兑现 (correctness): 三个适配器的 build_verbose_response 统一改为 language=None，并通过测试契约固定该行为。

风险与影响

- 风险：主要风险点如下：
- 子串匹配解析碰撞：resolve_adapter() 基于架构字符串子串匹配，新增 GlmAsr、GraniteSpeech、Qwen2Audio 键后，若未来出现名称相近的架构可能误配；测试覆盖了未知架构回退 Whisper，但真实模型名与注册键的映射仍需持续关注。
- Qwen2-Audio 30 秒静默截断：truncation=True 使长音频只转录前 30 秒，即使有 warning 日志，对生产用户仍是信息丢失；且该行为由 HF 处理器强约束，无法在适配器层绕过。
- 模型侧嵌入对齐重写：get_audio_feature 的 30s 窗口重组与 input_features_mask 降采样逻辑较复杂，若 conv 子采样或 merge_factor 与占位符 token 数失配，可能引发 token/embedding 数量不一致或崩溃；该路径缺乏 GPU e2e 测试。
- 多模态处理器回归面：qwen_audio.py、transformers_auto.py、base_processor.py 均在共享提示构造链路中改动，聊天路径与转录路径共用 processor，提示兜底逻辑的触发条件（空文本）若被其他入口误触发可能改变行为。
- 影响：影响范围集中在 OpenAI 转录入口、多模态处理器与 GLM-ASR 模型前向：
- 用户侧：Qwen2-Audio-7B、GLM-ASR-Nano、Granite Speech 3.3-8B 三类模型的 /v1/audio/transcriptions 从不可用 / 回退 Whisper 变为真实 ASR；时长缩放 max_new_tokens 避免长片段静默截断与短片段浪费生成预算。
- 系统侧：新增 3 个适配器文件、2 个 processor 提示兜底逻辑、1 个模型前向重写，并新增 CPU 单测套件；对既有 Whisper 路径不改变行为（Whisper 仍是默认回退）。
- 团队侧：确立了解码器语音模型转录适配器的扩展模式，后续新 ASR 模型可复用注册、契约测试与按时长算 token 预算的策略。

- 风险标记：适配器注册键子串匹配易碰撞，Qwen2-Audio 超过 30s 静默截断，模型侧嵌入对齐重写缺 GPU *e2e* 测试，共享多模态处理器提示链路回归面

关联脉络

- PR #34859 Qwen3.8-27B Model Support: 同属 SGLang 对 Qwen 系新模型的接入工作，且本 PR 的适配器设计沿用了仓库中 qwen3_asr 已有的适配器处理模式，体现同一功能线的迭代。
- PR #33518 feat(api): add sglext_spec: 同为 OpenAI 兼容入口层的协议扩展，与 /v1/audio/transcriptions 适配器注册共享 entrypoints/openai 模块，构成入口能力演进的相邻脉络。