PR #39674 完整报告

vllm-project/vllm

support hotwords for FunASR model

合并时间：2026-04-22 17:25

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/39674

执行摘要

- 一句话：为 FunASR 模型添加热词支持
- 推荐动作：值得精读，尤其是如何通过 SpeechToTextParams 数据类统一传递模型参数，以及如何在 prompt 构造中动态注入用户输入。设计上避免了修改 get_generation_prompt 的函数签名，扩展性良好。

功能与动机

FunASR 等 ASR 模型支持热词可显著提高领域术语、人名等专有词汇的识别效果。PR body 中作者给出了客户端调用示例，并关联了上游 PR #33247，表明该功能有明确用户需求。

实现拆解

1. 配置层（vllm/config/speech_to_text.py）：在 SpeechToTextParams 数据类中新增 hotwords: str | None 字段，作为模型 prompt 构造的输入。
2. API 协议层（vllm/entrypoints/openai/speech_to_text/protocol.py）：在 TranscriptionRequest 和 TranslationRequest 中均添加 hotwords 字段，并在 build_stt_params() 方法中将其传入 SpeechToTextParams。
3. 模型实现（vllm/model_executor/models/funasr.py）：在 FunASRModel.get_generation_prompt() 中读取 stt_params.hotwords，若不为空则构造包含热词列表的差异化 prompt（如“热词列表：[hello,world]”），否则使用默认 prompt。
4. 客户端示例（examples/online_serving/openai_transcription_client.py）：为同步和异步客户端函数添加 hotwords 参数，并通过 CLI --hotwords 传递。

关键文件：

- examples/online_serving/openai_transcription_client.py（模块 示例；类别 source；类型 core-logic；符号 stream_openai_response, sync_openai）：客户端示例，演示如何通过 CLI 参数传递热词，并展示同步 / 异步调用方式。
- vllm/config/speech_to_text.py（模块 配置层；类别 source；类型 data-contract；符号 SpeechToTextParams）：定义 SpeechToTextParams 数据类，新增 hotwords 字段作为模型 prompt 构造的输入。
- vllm/entrypoints/openai/speech_to_text/protocol.py（模块 协议层；类别 source；类型 core-logic；符号 TranscriptionRequest, TranslationRequest, build_stt_params）：API 协议层，为 TranscriptionRequest 和 TranslationRequest 添加 hotwords 字段，并在

build_stt_params 中传递。

- vllm/model_executor/models/funasr.py（模块 模型层；类别 source；类型 data-contract；符号 FunASRModel, get_generation_prompt）：模型实现层，在 get_generation_prompt 中读取 hotwords 并构造包含热词的 prompt 字符串。

关键符号：sync_openai, stream_openai_response, FunASRModel.get_generation_prompt, TranscriptionRequest.build_stt_params, TranslationRequest.build_stt_params

关键源码片段

examples/online_serving/openai_transcription_client.py

客户端示例，演示如何通过 CLI 参数传递热词，并展示同步 / 异步调用方式。

```
# 同步函数增加 hotwords 参数
async def stream_openai_response(
    audio_path: str, client: AsyncOpenAI, model: str, hotwords: str = None
):
    """
    Perform asynchronous transcription using OpenAI-compatible API.
    """
    print("\ntranscription result [stream]:", end=" ")
    with open(audio_path, "rb") as f:
        transcription = await client.audio.transcriptions.create(
            file=f,
            model=model,
            language="en",
            response_format="json",
            temperature=0.0,
            # 在 extra_body 中传递 hotwords，不改变 OpenAI 标准接口
            extra_body=dict(
                seed=420,
                top_p=0.6,
                hotwords=hotwords, # 新增热词参数
            ),
            stream=True,
        )
        async for chunk in transcription:
            if chunk.choices:
                content = chunk.choices[0].get("delta", {}).get("content")
                print(content, end="", flush=True)
    print() # Final newline after stream ends
```

vllm/entrypoints/openai/speech_to_text/protocol.py

API 协议层，为 TranscriptionRequest 和 TranslationRequest 添加 hotwords 字段，并在 build_stt_params 中传递。

```
class TranscriptionRequest(OpenAIBaseModel):
    # ... 其他字段
```

```
    hotwords: str | None = None
    """
    hotwords refers to a list of important words or phrases that the model
    should pay extra attention to during transcription.
    """

    def build_stt_params(self, audio, stt_config, model_config, task_type) ->
    SpeechToTextParams:
        return SpeechToTextParams(
            audio=audio,
            stt_config=stt_config,
            model_config=model_config,
            language=self.language,
            task_type=task_type,
            request_prompt=self.prompt,
            to_language=self.to_language,
            hotwords=self.hotwords, # 传递热词
        )

class TranslationRequest(OpenAIBaseModel):
    # ... 同样添加 hotwords 字段
    hotwords: str | None = None
    def build_stt_params(self, ...):
        return SpeechToTextParams(
            ...,
            hotwords=self.hotwords,
        )
```

vllm/model_executor/models/funasr.py

模型实现层，在 get_generation_prompt 中读取 hotwords 并构造包含热词的 prompt 字符串。

```
@classmethod
def get_generation_prompt(cls, stt_params: SpeechToTextParams) -> PromptType:
    audio = stt_params.audio
    stt_config = stt_params.stt_config
    language = stt_params.language
    hotwords = stt_params.hotwords # 新增：获取热词

    if language is None:
        raise ValueError("Language must be specified when creating the funasr prompt")

    # 根据热词是否存在，选择不同的 prompt 模板
    if hotwords is not None:
        funasr_prompt = (
            "<|im_start|>system\nYou are a helpful assistant.<|im_end|>\n"
            "<|im_start|>user\n请结合上下文信息，更加准确地完成语音转写任务。"
            "如果没有相关信息，我们会留空。

**上下文信息：**
```

```
"
            f"热词列表：[{hotwords}]\n语音转写：<|AUDIO|><|im_end|>\n"
            "<|im_start|>assistant\n"
        )
    else:
        funasr_prompt = (
            "<|im_start|>system\nYou are a helpful assistant.<|im_end|>\n"
            "<|im_start|>user\n语音转写：<|AUDIO|><|im_end|>\n"
            "<|im_start|>assistant\n"
        )

    prompt = {
        "prompt": funasr_prompt,
        "multi_modal_data": {
            "audio": (audio, stt_config.sample_rate),
        },
    }
    return cast(PromptType, prompt)
```

评论区精华

> gemini-code-assist[bot] 指出风险：在共享预处理函数 _preprocess_speech_to_text 中直接访问 request.hotwords，若请求为 TranslationRequest（当时未定义该字段）会引发 AttributeError，建议使用 getattr 安全访问。 结论：最终代码中 TranslationRequest 也添加了 hotwords 字段，该问题已解决。 DarkLight1337 要求：等待 #36268（SpeechToTextParams 重构）合并后再整合此功能，以确保新参数顺利接入。作者采纳了该建议，PR 在 #36268 合并后完成整合。

- 共享预处理函数中 hotwords 属性安全访问 (correctness): 开发者随后在 TranslationRequest 中也添加了 hotwords 字段，避免了 AttributeError，问题解决。

风险与影响

- 风险：
  1. 兼容性风险：最初 hotwords 仅添加到 TranscriptionRequest，而 TranslationRequest 未包含，导致共享预处理函数崩溃。已通过同时添加字段修复。
  2. 模型特定 prompt 格式：热词注入方式（直接格式化到 prompt 字符串）依赖 FunASR 模型的 prompt 结构，其他 ASR 模型（如 Whisper）可能不支持或需要不同格式。当前实现仅在 FunASRModel 中生效，其他模型未受影响。
  3. 缺少测试覆盖：PR 未包含单元测试或集成测试，热词功能在重构或回归中可能被破坏。- 影响：用户：可通过 API 的 extra_body 或客户端 --hotwords 参数传递逗号分隔的热词字符串，提升特定场景下的转录准确率。系统：新增可选字段，不影响现有请求；参数传递路径清晰，对性能无显著影响。团队：为后续其他 ASR 模型的热词支持提供了可复用的模式（通过 SpeechToTextParams 扩展）。
- 风险标记：跨请求类型兼容性，模型特定 prompt 格式，缺少测试覆盖

关联脉络

- PR #33247 original hotwords PR (upstream): PR body 中提及此 PR 为 #33247 的后续，表明该功能源自上游工作。
- PR #36268 Bundle get_generation_prompt() params into SpeechToTextParams: DarkLight1337 要求等待此 PR 合并后再整合热词参数，以确保新参数能通过 SpeechToTextParams 统一传递。本 PR 正是基于此重构实现。