PR #22997 完整报告

sgl-project/sglang

[Whisper] Automatic language detection via structured generation

合并时间：2026-04-27 15:54

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/22997

执行摘要

- 一句话：Whisper 自动语言检测：单次请求完成检测 + 转录
- 推荐动作：值得精读。该 PR 展示如何利用 SGLang 的结构化生成（regex）实现多阶段约束解码，将两步过程融合为单次请求。流式处理中的前缀缓冲 + 惰性发射模式设计精巧。adapter 基类接口设计为未来扩展提供模板。建议关注 parse_fused_output 的 fail-strict 策略、预热编译避免抖动、以及特殊令牌剥离时的精准性（只剥离已知 Whisper 令牌，避免破坏用户文本）。

功能与动机

当未提供 language 参数时，Whisper 服务器默认为英文（<|en|>），导致非英语音频输出错误。本 PR 使用 SGLang 的原生结构化生成，在单次请求中融合语言检测和转录，开销最小。参考 vLLM 的两阶段方法（vllm-project/vllm#34342）和 SGLang 的 #21190（Whisper CUDA 图优化）。

实现拆解

1. 基类接口扩展（transcription_adapters/base.py）：在 TranscriptionAdapter 中添加 supports_language_detection 属性、build_fused_autodetect_params、parse_fused_output 和 strip_special_tokens 静态方法，为其他 ASR 模型提供扩展点。
2. Whisper 适配器实现（transcription_adapters/whisper.py）：定义两个正则表达式变体（WHISPER_AUTODETECT_REGEX 无时间戳，WHISPER_AUTODETECT_TS_REGEX 带时间戳）。build_fused_autodetect_params 在 sampling_params 中设置 regex、skip_special_tokens=False 和 _detect_language 标志。parse_fused_output 解析输出文本，提取语言代码并剥离特殊令牌，失败时返回 (None, None)。strip_special_tokens 作为回退。语言代码集合 WHISPER_LANG_TOKEN_CODES 来自 transformers 的 LANGUAGES，自动跟踪新代码。
3. 服务层集成（serving_transcription.py）：在 create_transcription 中，当 language is None 且适配器支持检测时，设置 request._fused_autodetect = True。非流式处理调用 parse_fused_output 获取语言和透明文本；流式处理缓冲累积文本直到哨兵到达，然后发出已剥离前缀的 delta。build_verbose_response 不再默认 language='en'，直接传递检测结果（可能为 null）。
4. 预热编译（warmup.py）：新增 whisper_autodetect 预热函数，使用 0.1 秒静音音频生成 4 个 token 触发 xgrammar 编译两个正则变体的 FSM，避免首次请求的 ~15-20s 编译

抖动。

5. 多模态处理器适配（multimodal/processors/whisper.py）：检测 _detect_language 采样参数，将解码器提示改为仅 <|startoftranscript|>（1 token），使 FSM 约束后续 3 个 token 为语言、任务和 timestamps/notimestamps 令牌。
6. 测试配套：新增 test_whisper_adapter.py（25 个单元测试覆盖 parse_fused_output 的 happy path、边界、失败模式）、test_serving_transcription.py（流式 fused 路径单元测试，包含增量模式和错误帧）、扩展现有集成测试 test_serving_transcription.py（auto-detect 与显式英文对比、流式、时间戳）。

关键文件：

- python/sglang/srt/entrypoints/openai/transcription_adapters/whisper.py（模块 Whisper 适配器；类别 source；类型 core-logic；符号 supports_language_detection, build_fused_autodetect_params, parse_fused_output, strip_special_tokens）：核心实现文件：包含语言检测逻辑、正则表达式构建、解析输出、特殊令牌剥离。
- python/sglang/srt/entrypoints/openai/transcription_adapters/base.py（模块 适配器基类；类别 source；类型 dependency-wiring；符号 supports_language_detection, build_fused_autodetect_params, parse_fused_output, strip_special_tokens）：基类定义语言检测接口，确保其他 ASR 模型可扩展。
- python/sglang/srt/entrypoints/openai/serving_transcription.py（模块 转录服务；类别 source；类型 core-logic）：服务层入口，管理 fused 标志、流式与非流式处理统一调用 parse_fused_output。
- python/sglang/srt/entrypoints/warmup.py（模块 预热；类别 source；类型 dependency-wiring；符号 whisper_autodetect）：预热编译两个正则变体的 FSM，避免首次请求的 ~15-20s 编译开销。
- test/registered/unit/entrypoints/openai/test_whisper_adapter.py（模块 测试；类别 test；类型 test-coverage；符号 TestWhisperParseFusedOutput, test_happy_english, test_happy_non_english, test_missing_language_prefix_defers）：单元测试 parse_fused_output 的各种边界和失败模式，包括 happy path、缺失哨兵、未知语言、时间戳变体等。
- test/registered/unit/entrypoints/openai/test_serving_transcription.py（模块 测试；类别 test；类型 test-coverage；符号 _chunk, _MockTokenizerManager, init, generate_request）：单元测试流式 fused 路径，包括累积模式、增量模式、错误帧。

关键符号：build_fused_autodetect_params, parse_fused_output, strip_special_tokens, whisper_autodetect, _generate_transcription_stream, create_transcription

关键源码片段

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

核心实现文件：包含语言检测逻辑、正则表达式构建、解析输出、特殊令牌剥离。

```
# 关键常量：标记 fused 模式的采样参数键
FUSED_AUTODETECT_FLAG = "_detect_language"
```

```
# 从 transformers 的 LANGUAGES 字典动态获取所有 Whisper 语言代码
WHISPER_LANG_TOKEN_CODES: frozenset[str] = frozenset(LANGUAGES.keys())

# 构建语言前缀正则（排序保证 FSM 缓存可复用）
_LANG_ALT = "|".join(re.escape(c) for c in sorted(WHISPER_LANG_TOKEN_CODES))
_LANG_PREFIX = r"<\|(" + _LANG_ALT + r")\|>"

# 两个正则变体：无时间戳 / 带时间戳
WHISPER_AUTODETECT_REGEX = (
    _LANG_PREFIX + r"<\|transcribe\|>" + r"<\|notimestamps\|>" + r"[\s\S]*"
)
WHISPER_AUTODETECT_TS_REGEX = (
    _LANG_PREFIX + r"<\|transcribe\|>" + r"<\|0\.00\|>" + r"[\s\S]*"
)

@staticmethod
def parse_fused_output(
    text: str, *, ts_variant: bool = False
) -> tuple[Optional[str], Optional[str]]:
    """
    解析 fused 输出，返回 (language_code, user_visible_text)。
    若强制前缀未完整到达或解析失败，返回 (None, None)。
    """
    prefix_re = _FUSED_PREFIX_RE_TS if ts_variant else _FUSED_PREFIX_RE_NOTS
    m = prefix_re.match(text)
    if not m:
        logger.warning("parse_fused_output: forced prefix not locatable in %r", text)
        return (None, None)
    lang = m.group(1)
    if lang not in WHISPER_LANG_TOKEN_CODES:
        logger.warning("parse_fused_output: detected lang %r not in Whisper vocab", lang)
        return (None, None)
    # 去掉前缀，得到纯粹的用户可见文本
    visible = text[m.end():]
    # 剥离所有已知的特殊令牌（语言代码、控制令牌、时间戳）
    visible = _WHISPER_SPECIAL_TOKEN_RE.sub("", visible)
    visible = visible.strip()
    return (lang, visible)

@staticmethod
def strip_special_tokens(text: str) -> str:
    """回退清洗：剥离所有 Whisper 特殊令牌语法，不验证语义。"""
    return _WHISPER_SPECIAL_TOKEN_RE.sub("", text).strip()
```

评论区精华

@JustinTong0323 在首次 review 中指出了 4 个关键问题：(1) 流式路径未调用 `parse_fused_output`，导致强制前缀和特殊令牌泄漏；(2) `parse_fused_output` 解析失败时静默返回 `"en"`，无日志；(3) 缺少哨兵时直接输出文本，泄漏令牌；(4) 预热只消费第一个 yield

，可能未完全安装 FSM。作者 @shenxiul 针对每个问题提交了修复：流式统一解析、fail-strict 返回 (None, None)、预热改用 async for 完全消费生成器。

第二次 review 中，@JustinTong0323 又指出 (a) verbose_json 在解析失败时 build_verbose_response 仍默认 language="en"；(b) 流式结束前哨兵未到达时客户端无法区分静音和检测失败；(c) 几个注释和变量名问题。作者也一一修正：传递 language=None 使客户端可见 null；添加 SSE 错误帧；修正文档字符串和变量名。

一个关键的技术发现是：Whisper tokenizer 将 <|0.00|>（id 50365）解码为空字符串，导致时间戳变体的 fused 输出无法被 parse_fused_output 匹配。@JustinTong0323 本地复现并确认，最终通过拆分正则、添加 ts_variant 参数、仅匹配 <|lang|><|transcribe|> 解决。

- 流式路径泄漏特殊令牌 (correctness): @shenxiul 修复：流式处理现在使用同一 parse_fused_output 函数，在前缀完整到达前缓冲，发出时已清除令牌。
- 静默英文默认值 (correctness): 改为返回 (None, None) 并记录警告；调用者不再覆盖 request.language；verbose_json 中 language 字段为 null。
- 时间戳变体解码问题 (design): 通过拆分正则模式并添加 ts_variant 参数解决：时间戳变体仅匹配 <|lang|><|transcribe|>（忽略不可见的 <|0.00|>），并通过 output_ids 直接解析时间戳。
- 预热未完全消费生成器 (performance): 使用 async for _ in ...: pass 完全消费生成器，确保 FSM 编译完成且错误可被捕获。
- 流式结束前哨兵未到达 (correctness): 添加显式 SSE 错误帧 "language auto-detect failed: forced-prefix sentinel was not produced before stream end"，避免静默失败。

风险与影响

- 风险：
  1. 流式路径特殊令牌泄漏：已通过统一解析（parse_fused_output 处理前缀剥离和特殊令牌清除）和严格的 defer / error 机制解决。
  2. 语言检测失败静默回退：已改为 fail-strict：parse_fused_output 失败时返回 (None, None)，调用者不覆盖 request.language，verbose_json 返回 language: null。
  3. 预热编译启动时间增加：每增加 ~15-20s 启动时间（两个正则变体各一次），对于需要快速重启的场景可能不可接受，可通过跳过预热或配置移除。
  4. 正则表达式涵盖所有 Whisper 语言（100 种）：对旧版本 tokenizer 可能包含不在 vocab 中的代码，但 xgrammar 自动忽略不匹配分支，无影响。
  5. 对非 Whisper 模型无影响：通过 supports_language_detection 属性隔离，其他适配器返回 False，不会进入 fused 路径。
  6. TP>1 或分离编码器场景未验证：fe_kwargs["device"]="cuda" 已回退，建议后续 PR 专门处理。 - 影响：用户影响：非英语语音现在自动正确检测语言，转录质量显著提升。开发者可通过 language 参数显式指定或依赖自动检测。系统影响：运行中吞吐量降低约 1.1x（相比固定英语），但相比 vLLM 两阶段方法提升 5.8 倍。流式吞吐量因 SSE 框架开销固有降低约 3.7x（与是否检测无关）。团队影响：清晰的设计模式（adapter 接口 + fuse 策略），便于为其他 ASR 模型添加类似功能。新增 ~1285 行代码，~52 行删除，测试覆盖充分。

- 风险标记：预热增加启动时间，流式路径需重点测试，正则表达式维护负担

关联脉络

- 暂无明显关联 PR