PR #47189 完整报告

vllm-project/vllm

[Frontend] Cohere chat v2 api support

合并时间：2026-08-01 13:53

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/47189

执行摘要

- 一句话：新增 Cohere Chat v2 API 前端支持与 citations 渲染
- 推荐动作：值得精读。该 PR 是 vLLM 前端协议扩展的范例：通过子类化保持既有 OpenAI schema 稳定、以中间件统一外部 API 错误信封、用 env 门控控制新端点暴露面，以及用构造器钩子避免复制分支逻辑。建议关注 api_router.py 的错误翻译中间件、serving.py 的 SSE 状态机与 cohere_chat_message.py 的序列化技巧；同时提醒排查 /cohere 未纳入认证前缀的安全遗留问题。

功能与动机

RFC #43015 提出 vLLM 目前不支持 Cohere 的 /v2/chat HTTP API，而该 API 拥有 citations、thinking、tool_plan 等独特字段，重要用户希望直接对 vLLM 服务 Cohere 模型。RFC 建议仿照 create_chat_completion 流程增加 /cohere/v2/chat 入口，用 Melody 库完成 prompt 渲染，并保留已有的输出解析逻辑；本 PR 即按此路径落地。

实现拆解

1. 协议与数据契约：新增 vllm/entrypoints/cohere/protocol.py，复用官方 cohere SDK 的 ChatMessageV2、ToolV2、Document、Citation 等 wire 类型，本地只维护 CohereChatV2Request、非流式响应信封 CohereChatV2Response 和流式事件类型。通过 field_validator 完成 developer -> system 角色归一化、max_tokens 非负校验和 messages 非空校验；CohereError 统一为 {message, id} 结构。
2. 路由与门控：新增 vllm/entrypoints/cohere/api_router.py，模块加载时探测 cohere SDK，attach_router 同时受 VLLM_ENABLE_COHERE_API=1 与 SDK 可用性双重门控；未开启时跳过注册且区分 DEBUG/WARNING 日志。chat_v2 handler 返回三态：ErrorResponse 翻译为 CohereError、CohereChatV2Response 输出 JSON、其余视为 SSE 流。CohereErrorEnvelopeMiddleware 把全局异常处理器产生的 vLLM ErrorResponse 体改写为 Cohere 错误形状，保证 OpenAPI responses 与线上行为一致。
3. 服务层转换：新增 vllm/entrypoints/cohere/serving.py，CohereServingChatV2 继承 OpenAIServingChat，create_chat_v2 先经 _convert_v2_to_chat_completion 转换消息、tools、response_format、stream 选项，并把 Cohere 特有的 documents / safety_mode / citation_options / thinking 通过 chat_template_kwargs 透传给渲染器；非流式响应完成 finish_reason、usage 映射，流式响应用 _StreamState 状态机生成 message-start/content-start/content-delta/citation 事件并以 [DONE] 结束。

4. 渲染器：新增 vllm/renderers/cohere.py，CohereRenderer 用 cohere_melody Rust 绑定渲染 cmd3 / cmd4 模板，统一来自 CohereServingChatV2 与直接 chat_template_kwargs 调用者的两套词汇；_role_to_melody 完成 assistant -> chatbot、developer -> system 映射，_normalize_tool_call 把 OpenAI 工具调用转成 melody 的 {id, name, parameters} 形状，并在 _build_render_config 中拒绝 template_jinja / template 直传。
5. 消息扩展与推理解析：新增 vllm/entrypoints/cohere/cohere_chat_message.py，用 CohereChatMessage / CohereDeltaMessage 子类承载 citations 字段，序列化时剥离空 citations；vllm/reasoning/cohere_command_reasoning_parser.py 通过 POSITION_TO_SOURCE_KEY 与 MESSAGES_CITATIONS_KEY 把 melody 数值坐标解析成真实 document/tool source，并生成对应 citation。
6. 配套改动：OpenAIServingChat 增加 _create_chat_message 构造钩子，ChatCompletionResponseChoice 等改用 SerializeAsAny[ChatMessage]；vllm/envs.py 注册 VLLM_ENABLE_COHERE_API，CLI 增加 --cohere-is-reasoning-model 等参数；测试覆盖单元层（protocol、conversion、streaming、renderer、router）与 E2E 层（轻量 SmolLM2-135M-Instruct 起真实 server，含 Cohere SDK 回环）。

关键文件：

- vllm/entrypoints/cohere/api_router.py（模块 请求路由；类别 source；类型 entrypoint；符号 attach_router, chat_v2, CohereErrorEnvelopeMiddleware, _translate_vllm_error_body）：新增 /cohere/v2/chat 的注册与门控逻辑：模块加载时探测 cohere SDK，attach_router 同时受 VLLM_ENABLE_COHERE_API 控制；路由 handler 负责把 vLLM 内部 ErrorResponse 翻译成 CohereError，并用中间件兜底全局异常。
- vllm/entrypoints/cohere/serving.py（模块 服务层；类别 source；类型 core-logic；符号 CohereServingChatV2, create_chat_v2, _convert_v2_to_chat_completion, _chat_completion_to_v2）：核心服务层：CohereServingChatV2 继承 OpenAIServingChat，负责 v2 请求到 ChatCompletionRequest 的转换、非流式响应映射、SSE 流状态机（_StreamState）以及 citations 的 wire 形状转换。
- vllm/entrypoints/cohere/protocol.py（模块 协议层；类别 source；类型 data-contract；符号 CohereChatV2Request, CohereChatV2Response, CohereError, CohereFinishReason）：协议契约：复用 cohere SDK 各消息 / 工具 / 文档类型，本地只维护请求体、非流式响应信封和流式事件联合；包含 developer->system 的角色归一化与 max_tokens 校验。
- vllm/renderers/cohere.py（模块 渲染器；类别 source；类型 core-logic；符号 CohereRenderer, _role_to_melody, _normalize_tool_call, _content_blocks）：新增 CohereRenderer，用 cohere_melody Rust 绑定渲染 cmd3/cmd4 模板，统一来自 CohereServingChatV2 和直接 chat_template_kwargs 调用者的双词汇；安全拒绝 template_jinja/template 直传。
- vllm/entrypoints/cohere/cohere_chat_message.py（模块 消息扩展；类别 source；类型 core-logic；符号 Citation, CitationSource, CohereChatMessage, CohereDeltaMessage）：以子类方式为 ChatMessage/DeltaMessage 增加 citations 字段，并通过 SerializeAsAny 在响应信封中保留子类 schema，避免污染 OpenAI 协议。

- vllm/entrypoints/openai/chat_completion/serving.py（模块 服务层；类别 source；类型 core-logic；符号 _create_chat_message, _finalize_response_message）：为 OpenAIServingChat 增加 _create_chat_message 构造钩子并调整 _finalize_response_message，使子类可以替换响应 ChatMessage 类型而不复制分支逻辑。
- vllm/reasoning/cohere_command_reasoning_parser.py（模块 推理解析；类别 source；类型 core-logic；符号 _melody_sources_to_vllm, _melody_citations_to_vllm）：推理解析器通过 POSITION_TO_SOURCE_KEY 和 MESSAGES_CITATIONS_KEY 把 melody 数值坐标解析成真实 document/tool 源，并生成 Citation/CohereDeltaMessage。
- tests/entrypoints/cohere/test_chat_v2.py（模块 集成测试；类别 test；类型 test-coverage；符号 test_cohere_v2_chat_non_streaming, test_cohere_v2_chat_streaming, test_cohere_v2_chat_validation_error_returns_400, test_cohere_v2_chat_documents_field_accepted）：E2E 集成测试：CPU 可跑的真实 serve 进程 + httpx 覆盖非流式 / 流式 / 校验错误 /documents 字段，并用官方 cohere SDK 做回环；验证路由、转换和 SSE 生命周期。

关键符号：attach_router, chat_v2, CohereErrorEnvelopeMiddleware.dispatch, CohereServingChatV2.create_chat_v2, CohereServingChatV2._convert_v2_to_chat_completion, CohereServingChatV2._chat_completion_to_v2, CohereServingChatV2._chat_completion_stream_to_v2, CohereServingChatV2._apply_cohere_template_kwargs, CohereRenderer.render, _role_to_melody, _normalize_tool_call, CohereChatMessage._serialize, CohereDeltaMessage._serialize, CohereChatV2Request._normalize_message_roles, OpenAIServingChat._create_chat_message, _melody_sources_to_vllm

关键源码片段

vllm/entrypoints/cohere/serving.py

核心服务层：CohereServingChatV2 继承 OpenAIServingChat，负责 v2 请求到 ChatCompletionRequest 的转换、非流式响应映射、SSE 流状态机（_StreamState）以及 citations 的 wire 形状转换。

```
class CohereServingChatV2(OpenAIServingChat):
    """Cohere Chat v2 服务处理器。

    核心思路：把 Cohere v2 请求先翻译成 OpenAIServingChat 能识别的
    ChatCompletionRequest，再复用整条 chat completion 生成链路；
    Cohere 特有的 documents / safety_mode / citation_options 等字段
    通过 chat_template_kwargs 透传给渲染器。
    """

    async def create_chat_v2(
        self,
        request: CohereChatV2Request,
        raw_request: Request | None = None,
    ) -> AsyncGenerator[str, None] | CohereChatV2Response | ErrorResponse:
        # 先做 v2 -> OpenAI 的字段映射（messages、tools、stream 等），
```

```
        # 再整体下发，保持与 create_chat_completion 一致的行为。
        chat_req = self._convert_v2_to_chat_completion(request)
        generator = await self.create_chat_completion(chat_req, raw_request)

        # 三种出口：内部错误原样返回；普通响应转成 Cohere v2 信封；
        # 流式响应转成 message-start / content-* / message-end 事件序列。
        match generator:
            case ErrorResponse():
                return generator
            case ChatCompletionResponse():
                return self._chat_completion_to_v2(generator, request)
            case _:
                return self._chat_completion_stream_to_v2(generator, request)

    @classmethod
    def _convert_v2_to_chat_completion(
        cls, request: CohereChatV2Request
    ) -> ChatCompletionRequest:
        # 逐条转换消息（user / assistant / system / tool），
        # 然后按功能分组调用子转换器，保持单一职责。
        openai_messages: list[dict[str, Any]] = []
        cls._convert_messages(request.messages, openai_messages)
        chat_req = cls._build_base_chat_completion(request, openai_messages)
        cls._apply_streaming_options(chat_req, request)
        cls._apply_response_format(chat_req, request)
        cls._apply_tools(chat_req, request)
        cls._apply_tool_choice(chat_req, request)
        cls._apply_cohere_template_kwargs(chat_req, request)
        return chat_req
```

vllm/entrypoints/cohere/cohere_chat_message.py

以子类方式为 ChatMessage/DeltaMessage 增加 citations 字段，并通过 SerializeAsAny 在响应信封中保留子类 schema，避免污染 OpenAI 协议。

```
class CitationSource(OpenAIBaseModel):
    """citation 的源归属（document / tool）。

    由 Cohere 推理解析器在生成时就解析出真实 document / tool id，
    serving 层只负责转成 SDK 的 cohere.types.Citation。
    """
    type: Literal['document', 'tool'] | None = None
    id: str | None = None
    document: dict[str, Any] | None = None
    tool_output: dict[str, Any] | None = None


class CohereChatMessage(ChatMessage):
    """带 citations 的 ChatMessage 子类。
```

```
    只有 CohereServingChatV2 走这条路径；普通 OpenAI 响应仍是
    原版 ChatMessage，schema 不受影响。响应信封里声明了
    SerializeAsAny[ChatMessage]，所以子类字段能保留到 JSON。
    """
    citations: list[Citation] | None = None

    @model_serializer(mode='wrap')
    def _serialize(self, handler):
        # mode='wrap' 会整体覆盖父类的 model_serializer，因此显式
        # 调用 super()._serialize 保留父类清理逻辑（如剥离空 tool_calls）；
        # 再删掉未设置的 citations，让可选扩展省略而不是输出 null。
        data = super()._serialize(handler)
        if not data.get('citations'):
            data.pop('citations', None)
        return data
```

评论区精华

核心争论在于是否让 OpenAI 协议承载 Cohere 专属字段：DarkLight1337 认为不应为覆盖其他 API 而扩展 OpenAI 协议，建议抽取公共代码；作者改为 ChatMessage 子类 + SerializeAsAny，并承诺由 Cohere 承担维护。另一个争议是新端点默认暴露面：chaunceyjiang 建议新增端点必须默认关闭，最终引入 VLLM_ENABLE_COHERE_API 环境变量优化。depthfirst-app[bot] 提出 chat_template_kwargs['template_jinja'] 绕过 trust_request_chat_template 的注入风险，最终渲染器拒绝非 None 直传。aarnphm 与 sfeng33 讨论 parser 的 adjust_response 钩子，认为应把 citations 元数据从 serving 类移出，属于非阻塞后续项。

- 是否将 citations 字段加入 OpenAI 协议 (design): 采用 CohereChatMessage/CohereDeltaMessage 子类 + SerializeAsAny 序列化，基础 ChatMessage/DeltaMessage schema 不变，由 Cohere 承担后续维护。
- 新端点默认开启还是 opt-in (design): /cohere/v2/chat 仅在 VLLM_ENABLE_COHERE_API=1 且安装 cohere SDK 时注册，默认不暴露。
- 认证保护缺失：/cohere 不在 GUARDED_PREFIX (security): PR 内未修改 server_utils.py；端点默认关闭缓解了暴露面，但开启后的认证覆盖问题未在本 PR 内解决，需后续跟进。
- chat_template_kwargs 的 template_jinja 注入面 (security): 渲染器最终把 template_jinja/template 纳入消费键集，并在 _build_render_config 拒绝非 None 值，注入面被收敛。
- parser 的 adjust_response 钩子 (design): 决定本 PR 不实施，Cohere 侧后续跟进；当前通过 _create_chat_message 钩子 + ChatMessage 子类解决。
- 疑似 AI 生成的冗余测试 (testing): 删除冗余测试，保留有区分度的用例。

风险与影响

- 风险：安全方面存在两个关注点：一是 /cohere/v2/chat 不在 AuthenticationMiddleware 的 GUARDED_PREFIX（/v1、/v2、/inference）内，若运维开启

VLLM_ENABLE_COHERE_API 且配置 --api-key，该端点可能未受认证保护；本 PR 未修改 server_utils.py，需后续跟进。二是 chat_template_kwargs 由用户控制，虽然 template_jinja 已被拒绝，但 setdefault 允许用户覆盖 documents 等字段只影响自身请求，风险可控。兼容性方面，OpenAIServingChat 新增 _create_chat_message 钩子并调整 _finalize_response_message，默认返回值不变，但对带 reasoning 的输出路径有轻微改动，需要依赖输出形状的调用方注意。流式转换的 _StreamState 状态机复杂度高，若 thinking 与 tool_call 交错出现而块未正确闭合，可能产生非法事件序列，已有单测覆盖但仍需真实模型验证。

- 影响：对用户：使用 Cohere Command 系列模型的用户可直接调用 /cohere/v2/chat，获得 citations、thinking、工具调用等原生语义；非 Cohere 部署默认不受影响。对系统：端点默认关闭，只有显式设置 VLLM_ENABLE_COHERE_API=1 且安装 cohere SDK 时才注册路由与中间件；中间件只作用于 /cohere/* 前缀的错误响应，不影响其他 API。对团队：OpenAI serving 层新增一个扩展点，后续其它非 OpenAI 协议可复用，但 OpenAI 协议也由此引入了 SerializeAsAny 的子类序列化机制，schema 复杂度小幅上升；Cohere 侧承诺跟进 parser adjust_response 等后续重构。
- 风险标记：认证前缀遗漏待跟进，OpenAI 核心服务钩子变更，SSE 状态机复杂度高，可选依赖缺失时入口静默关闭，模板注入面已收敛

关联脉络

- PR #50334 [Bugfix][Responses] Add tests for Chat Completions Responses API Render Parity: 同属前端协议与渲染一致性工作线，涉及 ChatMessage 序列化、工具渲染和 API 间 parity 测试，与本 PR 的渲染器与消息扩展改动相互印证。
- PR #50642 [Bugfix][Parser] Forward model_config to nested reasoning parsers: 与 vllm/reasoning 解析器相关，本 PR 同样调整了 Cohere 推理解析器的 source/citation 解析逻辑。
- PR #50515 [ROCm][CI] Restore Mistral tool-parser compatibility after unification: 同样触及前端工具解析与 OpenAI serving 层，体现工具调用路径的持续演进。