PR #47735 完整报告

vllm-project/vllm

[Rust Frontend][CI] Unblock more end-to-end test cases

合并时间：2026-07-07 16:27

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/47735

执行摘要

本 PR 解锁了 Rust 前端 CI 中更多的端到端测试，为此调整了非流式响应中未设置字段的序列化方式，确保显式序列化为 null 以匹配 Python 后端行为；同时修复了 CLI 参数标记并扩展了 CI 配置（包括 AMD 平台）。

功能与动机

PR body 指出："Unblock some more end-to-end test cases in CI for Rust frontend as we're filling more gaps. This PR only enables the test cases that require no or minimal code and test changes." 为了填补 Rust 前端与 Python 后端的行为差异，需要在非流式响应中正确表示 null 字段，并且将 CLI 参数中已支持的项标记为 Noop 以免用户困惑。

实现拆解

1. 响应序列化约定变更：移除多个响应结构体上的 #[serde_with::skip_serializing_none]，使非流式响应中未设置的字段序列化为显式 null。影响文件包括 chat_completions/types.rs、openai/completions/types.rs、utils/types.rs、inference/generate/types.rs 等。
2. tool_calls 字段类型调整：将 ChatCompletionMessage 中的 tool_calls 从 Option<Vec<ToolCall>> 改为 Vec<ToolCall>，并使用 #[serde(skip_serializing_if = "Vec::is_empty")]，以匹配 Python 端空数组被弹出的行为（chat_completions/types.rs 及 convert.rs）。
3. CLI 参数修复：将 enable_tokenizer_info_endpoint 的类型从 Unsupported 改为 Noop 并添加 hide = true，因为 Rust 前端尚未实现 /tokenizer_info 端点（unsupported.rs）。
4. 测试断言增强：在 tests.rs 的 non_stream_chat_returns_json_response 和 non_stream_completions_return_json_response 等测试中新增对 null 字段存在性和值的断言，确保序列化行为符合预期。
5. CI 配置扩展：在 .buildkite/test_areas/rust_frontend.yaml 和 .buildkite/test-amd.yaml 中添加更多端到端测试条目，使 CI 覆盖更全面。

rust/src/server/src/routes/tests.rs

新增对非流式响应 null 字段的断言，是验证序列化约定变更的核心测试文件。

```
// 在 non_stream_chat_returns_json_response 测试中，验证未设置字段必须为显式 null
let response_object = json.as_object().expect("response object");
let choice = json["choices"][0].as_object().expect("choice object");
```

```
let message = choice["message"].as_object().expect("message object");

// 遍历需要检查 null 的字段列表
for (object, key) in [
    (response_object, "system_fingerprint"),
    (response_object, "prompt_token_ids"),
    (response_object, "kv_transfer_params"),
    (choice, "logprobs"),
    (choice, "stop_reason"),
    (choice, "token_ids"),
    (message, "reasoning"),
] {
    // 断言字段存在且值为 null
    assert!(
        object.contains_key(key) && object[key].is_null(),
        "expected explicit null `{key}`: {json}"
    );
}
// 而 tool_calls 在 Python 端为空时会被弹出，所以不应存在于响应中
assert!(!message.contains_key("tool_calls"), "{json}");
```

rust/src/cmd/src/cli/unsupported.rs

修改 enable_tokenizer_info_endpoint 参数类型和可见性，使其作为 Noop 并隐藏，避免用户误用。

```
/// Enable the `/tokenizer_info` endpoint. May expose chat
/// templates and other tokenizer configuration.
///
/// Accepted as a no-op: the Rust frontend serves `/tokenize` and
/// `/detokenize`, but does not implement `/tokenizer_info` yet.
#[arg(
    long,
    visible_alias = "no-enable-tokenizer-info-endpoint",
    default_missing_value = "true",
    num_args = 0..=1,
    hide = true // 将参数隐藏，防止用户误以为支持
)]
pub enable_tokenizer_info_endpoint: Option<Noop>, // 类型从 Unsupported 改为 Noop
```

rust/src/server/src/routes/openai/chat_completions/types.rs

移除了响应结构体上的 skip_serializing_none，并调整 tool_calls 类型，是序列化约定变更的核心。

```
/// Mirrors the Python vLLM `ChatCompletionResponse` class.
///
/// Do not skip serializing `None` fields here: non-streaming response types
/// should serialize `None` as explicit `null`.
#[derive(Debug, Clone, Serialize)] // 移除了 skip_serializing_none
pub(super) struct ChatCompletionResponse {
```

```
    pub id: String,
    pub object: String,
    pub created: u64,
    pub model: String,
    pub choices: Vec<ChatCompletionChoice>,
    pub usage: Option<Usage>,
    pub system_fingerprint: Option<String>,
    pub prompt_logprobs: Option<Vec<Option<HashMap<String, f32>>>>,
    pub prompt_token_ids: Option<Vec<u32>>,
    pub kv_transfer_params: Option<Value>,
}

/// Mirrors the Python vLLM response `ChatMessage` class.
#[derive(Debug, Clone, Serialize)]
pub(super) struct ChatCompletionMessage {
    pub role: AssistantRole,
    pub content: Option<String>,
    #[serde(skip_serializing_if = "Vec::is_empty")] // 空数组被跳过，匹配 Python 端行为
    pub tool_calls: Vec<ToolCall>, // 从 Option<Vec<ToolCall>> 改为 Vec<ToolCall>
    pub reasoning: Option<String>,
}
```

评论区精华

> AndreasKaratzas: “Can we reflect these in test-amd.yaml too? We just merged #47478”

作者在后续提交中处理了合并冲突，test-amd.yaml 已同步更新，确保 AMD CI 也运行 Rust 前端测试。

风险与影响

- 序列化兼容性：非流式响应现在包含显式 null 字段，依赖字段省略的客户端可能解析失败。但测试覆盖了主流响应路径，且此行为与 OpenAI API 规范一致。
- CLI 参数隐藏：enable_tokenizer_info_endpoint 被设为 Noop 并隐藏，用户可能误以为参数无效，但文档注释已说明原因，影响很小。
- CI 配置同步：test-amd.yaml 的更新已在冲突解决中完成，需合并后验证是否完整。

关联脉络

- PR #47478([ROCm][CI] Adding Rust parity）：本 PR 在其基础上扩展了 AMD CI 配置。
- PR #46768([Frontend] add per-request timing metrics field）：同样涉及响应结构体变更，本 PR 的序列化约定调整会影响 metrics 字段的呈现方式。
- 这些 PR 共同推进了 Rust 前端的稳定性与功能完备性，后续将继续解锁需要更大改动范围的测试用例。