PR #44391 完整报告

vllm-project/vllm

[Rust Frontend] Support include_reasoning=false

合并时间：2026-06-05 16:47

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/44391

执行摘要

本 PR 为 Rust 前端补全了 include_reasoning=false 的支持，使客户端能够选择不暴露模型推理链。核心改动涉及请求验证、参数传递、非流式与流式响应中的条件抑制，并修复了 logprobs 可能泄露隐藏推理 token 的安全问题。

功能与动机

关联 issue #44280 定义了 Rust 前端的特征路线图，其中 include_reasoning=false 是 OpenAI 聊天补全 API 的标准参数。Python 前端已支持，Rust 前端需要补齐此功能，以使用户在调用推理模型（如 Qwen 等）时能够控制是否显示 reasoning 内容。

实现拆解

1. 移除验证阶段拒绝：在 validate.rs 的 validate_request_compat 函数中，删除了原来对 include_reasoning=false 的拒绝检查块（if !request.include_reasoning { bail! }），使得该参数能够通过兼容性验证。同时新增了单元测试 validate_request_compat_accepts_include_reasoning_false 验证这一行为。
2. 传递参数到准备阶段：在 convert.rs 中，向 PreparedRequest 结构体新增 include_reasoning: bool 字段，并在 prepare_chat_request 函数中将 request.include_reasoning 的值拷贝到该字段。单元测试 prepare_chat_request_preserves_include_reasoning_false 确保当请求传 false 时，准备后的请求字段也为 false。
3. 非流式响应的条件抑制：在 chat_completions.rs 的 collect_chat_completion 函数中，首先获取 message.reasoning()，然后计算 include_output_metadata = include_reasoning || reasoning.is_none()。该条件用于控制：
   - reasoning 字段：仅在 include_reasoning 为 true 时输出。
   - logprobs：仅在 requested_logprobs && include_output_metadata 时输出，防止隐藏推理通过逐 token logprobs 泄露。
   - token_ids：仅在 return_token_ids && include_output_metadata 时输出。 这样，当 include_reasoning=false 且模型实际产生了 reasoning 时，与其关联的逐 token 元数据被一并隐藏。
4. 流式响应的条件抑制：在 chat_completion_chunk_stream 函数中，引入了局部变量 inside_hidden_reasoning 和 suppress_current_update_metadata 来跟踪状态。当遇到 BlockDelta(Reasoning) 时，若 include_reasoning=false，则跳过该增量块；同时标记

suppress_current_update_metadata，使得随后到来的 LogprobsDelta 和 token_ids 被丢弃，避免单独发出包含隐藏推理 token 元数据的 chunk。对 BlockStart/BlockEnd 中的 Reasoning 块，也做同样抑制，并正确处理 delimiter-only 块。

5. 测试覆盖：新增了：
   - include_reasoning_false_suppresses_reasoning_in_non_stream_chat：验证非流式响应中没有 reasoning 键。
   - include_reasoning_false_suppresses_non_stream_output_metadata：使用 mock engine 输出带 logprobs 的 token，验证 logprobs 和 token_ids 被省略。
   - 另外三个流式测试（在源码符号中可见）覆盖了 delta.reasoning 被抑制、logprobs 对被抑制的推理块跳过、以及 delimiter-only 块时 logprobs 也被跳过的场景。

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

核心逻辑文件，实现非流式和流式响应中 include_reasoning=false 的抑制逻辑，包括 include_output_metadata 条件和流式状态管理。

```
// 从引擎收集的 assistant 消息中提取 reasoning 内容
let reasoning = message.reasoning();

// 当用户指定了 include_reasoning=false 且消息包含 reasoning 时，
// 不仅隐藏 reasoning 字段，还必须隐藏关联的逐 token 元数据，
// 例如 logprobs 和 token_ids，防止信息泄露。
let include_output_metadata = include_reasoning || reasoning.is_none();

// 仅在 include_output_metadata 为 true 时输出 logprobs
let logprobs = if requested_logprobs && include_output_metadata {
    Some(decoded_logprobs_to_openai_chat(
        logprobs.as_ref().ok_or_else(|| {
            server_error!("chat response requested logprobs but generation returned none")
        })?,
        return_tokens_as_token_ids,
    )?)
} else {
    None
};

// 构造最终响应时，reasoning 字段受 include_reasoning 直接控制
ChatCompletionResponse {
    choices: vec![ChatCompletionResponseChoice {
        reasoning: if include_reasoning { reasoning } else { None },
        logprobs,
        token_ids: (return_token_ids && include_output_metadata).then_some(token_ids),
        // ... 其他字段
    }],
    // ... 其他字段
}
```

评论区精华

BugenZhao 关于变量命名与设计：

> “使用正面命名语义更清晰。添加一些注释解释行为。”“建议将 suppressed_reasoning 维护为局部变量，这样 PendingChunk 不需要感知 include_reasoning，可以统一两个路径。”

提交者采纳了建议，修改为 include_output_metadata 并将隐藏 reasoning 抑制状态移入流式处理函数的局部变量，PendingChatChunk 不再依赖 include_reasoning。

Codex 自动审查发现 logprobs 泄露风险（P1 级别）：

> “当 stream=true, include_reasoning=false, logprobs=true 时，此分支只丢弃了 reasoning delta，但随后的 LogprobsDelta 仍被缓冲并作为一个只含 logprobs 的 chunk 刷新，因此 choices[].logprobs.content[].token 可能包含隐藏的推理 token 文本。这违背了请求的抑制意图。”

深度优先安全审查也指出非流式路径中同样问题。提交者通过添加 suppress_current_update_metadata 条件修复，并增加了对应的单元测试覆盖。

风险与影响

风险：

- 信息泄露：若 logprobs 或 token_ids 过滤不完整，隐藏的推理 token 可通过逐 token 元数据泄露。当前已通过 include_output_metadata 和流式中的 suppress_current_update_metadata 解决，但仍需持续关注其他可能的元数据字段（如自定义结构化输出）。
- 流式边缘情况：当推理块仅包含分隔符（如单独的 <think>），需要确保其元数据也被抑制。测试已覆盖，但生产环境中的不同 tokenizer 行为可能引入未预见的情况。
- 行为一致性：Rust 前端与 Python 前端在 include_reasoning=false 下的行为必须一致，目前 PR 声称匹配，但缺乏详细的交叉对比测试。

影响：

- 用户：现在可以在 Rust 前端使用 include_reasoning=false，控制是否显示推理内容。
- 系统：无向后兼容性问题，默认行为不变。性能开销极小。
- 团队：后续需持续维护与 Python 前端的行为对齐，并在新增推理模型时确保相容性。

关联脉络

本 PR 是 Rust 前端特征路线图（issue #44280）的一部分，与同仓库中其他 Rust 前端 PR（如 #44591 批量自动中止、#44500 语言模型模式标志）一起逐步补齐与 Python 前端的差距。此外，该 PR 的测试模式与近期添加的 include_reasoning 相关测试（如 PR #43150）呼应，共同提升了推理功能在 Rust 前端的完整性。