PR #49604 完整报告

vllm-project/vllm

[Rust Frontend] Add --limit-mm-per-prompt support

合并时间：2026-07-29 17:21

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/49604

执行摘要

本 PR 为 Rust 前端添加 --limit-mm-per-prompt 支持，允许对每个请求中的图像、音频、视频等模态输入数量进行上限控制。未指定模态默认不设限。验证在 media 获取之前执行，超限请求返回 HTTP 400。该功能与 Python 前端行为对齐，并支持托管引擎模式。

功能与动机

为了与 Python 前端的行为对齐，Rust 前端需要支持 --limit-mm-per-prompt，以限制多模态输入数量，防止滥用或超过模型支持上限。Python 已有类似实现（vllm/config/multimodal.py 中的 limit_per_prompt 和 validate_num_items），Rust 前端需要补充这个能力。

实现拆解

1. 定义数据类型：在 rust/src/chat/src/multimodal.rs 新增 MmLimitModality 枚举（Image、Audio、Video）、MmLimitSpec 枚举（支持纯计数或带额外选项的对象）和类型别名 MmLimitPerPrompt。使用 serde 反序列化，并通过 flatten 保留未知字段转发引擎。
2. 注入 CLI 参数：在 rust/src/cmd/src/cli.rs 的 SharedRuntimeArgs 中添加 limit_mm_per_prompt 字段，通过 parse_json<MmLimitPerPrompt> 解析 JSON，并提供 limit_mm_per_prompt_json 方法序列化用于转发。
3. 传递到多模态模型信息：修改 MultimodalModelInfo::from_paths 签名，接收 MmLimitPerPrompt 并存储。通过多级构造函数传递到顶层。
4. 请求验证：实现 validate_mm_limits 函数，在 fetch_media 之前统计每个模态的 item 数量，与限制比较，超限返回错误。
5. 托管引擎转发：在 ManagedEngineArgs::into_config 中，当限制非空时，将 JSON 字符串以 --limit-mm-per-prompt 参数传递给 Python 引擎子进程。
6. 测试覆盖：cli/tests.rs 测试无效模态键被拒绝；routes/tests.rs 集成测试验证双图像请求在 limit=1 时返回 400。
7. 配套更新：更新示例 external_engine_openai_qwen.rs，调整序列化逻辑，移除过期配置。

rust/src/chat/src/multimodal.rs

核心变更文件，定义了多模态限制的数据结构、序列化和验证逻辑。

```
/// Per-modality item-count limits configured by `--limit-mm-per-prompt`.
///
```

```
/// Modalities absent from the map are unlimited.
pub type MmLimitPerPrompt = HashMap<MmLimitModality, MmLimitSpec>;

/// Modalities that `--limit-mm-per-prompt` can be keyed by.
#[derive(Debug, Clone, Copy, PartialEq, Eq, Hash, Serialize, Deserialize)]
#[serde(rename_all = "snake_case")]
pub enum MmLimitModality {
    Image,
    Audio,
    Video,
}

impl MmLimitModality {
    /// The wire name, matching Python's modality strings.
    pub fn as_str(self) -> &'static str {
        match self {
            Self::Image => "image",
            Self::Audio => "audio",
            Self::Video => "video",
        }
    }
}

/// One modality's limit, in either of the two shapes Python accepts.
#[derive(Debug, Clone, PartialEq, Eq, Serialize, Deserialize)]
#[serde(
    untagged,
    expecting = "an item count, or an object with an optional `count` field"
)]
pub enum MmLimitSpec {
    /// Legacy form: `"image": 16`
    Count(usize),
    /// Configurable form: `"video": {"count": 1, "num_frames": 32}`
    Options {
        /// Absent means unlimited, matching an absent modality.
        #[serde(default, skip_serializing_if = "Option::is_none")]
        count: Option<usize>,
        /// Preserve Python-owned options for forwarding. Never interpreted
        /// here: they size the engine's dummy-profiling encoder cache, which
        /// has no Rust counterpart.
        #[serde(flatten)]
        extra: BTreeMap<String, serde_json::Value>,
    },
}

impl MmLimitSpec {
    /// The configured item count, or `None` when this modality is unlimited.
    pub fn count(&self) -> Option<usize> {
        match self {
```

```
            Self::Count(count) => Some(*count),
            Self::Options { count, .. } => *count,
        }
    }
}
```

评论区精华

- 默认值设计（@BugenZhao）：*"I feel a limit of 999 doesn't make too much sense... Shall we default to unlimited instead?"* → 采纳，改为无限制。
- 类型安全（@BugenZhao）：推荐使用封闭枚举 MmLimitModality 代替 HashMap<String, usize> → 采用。
- 可配置形式支持（@chatgpt-codex-connector）：提出应支持对象形式以兼容 Python 协议 @cinnamonica02 回应当前保留 extra 字段转发，后续跟进。
- 模态合并（@BugenZhao 询问 ImageEmbeds 是否独立模态）：确认后合并计数。
- 托管引擎转发（@BugenZhao 确认参数是否被引擎读取）：确认 Python 端支持。

风险与影响

- 模态覆盖不完整：新增模态需同步更新枚举，否则静默忽略。
- 序列化兼容性：untagged 枚举可能失败，extra 字段未验证。
- 托管传递路径：参数传递安全但需注意序列化。
- 测试覆盖不足：仅测试图像模态，缺少音频视频边界测试。
- 向后兼容：默认无限制，无 breaking change。

关联脉络

本 PR 是 Rust 前端与 Python 前端功能对齐的延续。此前已有 PR #48145 复用 prefill token ids，PR #50093 添加 Kimi K3 前端等。本 PR 补全多模态限制功能，确保 Rust 前端在媒体管理上逐步与 Python 侧平齐。