PR #43463 完整报告

vllm-project/vllm

[Frontend] Expose logprob_token_ids on Python OpenAI endpoints

合并时间：2026-07-14 05:40

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/43463

执行摘要

- 一句话：在 OpenAI 端点暴露 logprob_token_ids 支持
- 推荐动作：值得精读，尤其是协议层字段添加、验证器编写和采样参数传递的模式。这是 vLLM 前端扩展规范化的良好示例。

功能与动机

Issue #43466 指出 OpenAI 聊天 / 补全端点无法接受 logprob_token_ids，但此功能对于需要为固定小词汇表（如多标签评分中的已知词汇 ID）获取 logprobs 的调用方非常有用。该能力已存在于底层 Python API 和 /generative_scoring 端点，但未暴露到默认的 OpenAI 兼容前端。

实现拆解

实现分为以下步骤：

1. 协议层字段定义：在 vllm/entrypoints/openai/chat_completion/protocol.py 的 ChatCompletionRequest 和 BatchChatCompletionRequest，以及 vllm/entrypoints/openai/completion/protocol.py 的 CompletionRequest 中，新增 logprob_token_ids: list[int] | None 字段，配以详细描述。
2. 采样参数传递：在 to_sampling_params() 方法中增加 logprob_token_ids=self.logprob_token_ids or None 的传递；同时调整 logprobs 的取值逻辑：当显式 ID 存在时，logprobs 设为 None 以阻止自然 top-k 路径，从而让引擎走显式 token ID 收集路径。
3. 请求验证：在协议类的 check_logprobs 和 check_batch_mode 验证器中添加三项检查：
   - logprob_token_ids 与 use_beam_search 互斥（beam search 评分路径不支持显式 ID）。
   - logprob_token_ids 要求已设置 logprobs。
   - 对 Completion 端额外要求：若 echo=True 且 max_tokens=0（无生成 token），则拒绝显式 ID 请求。
4. 服务层适配：在 vllm/entrypoints/openai/chat_completion/serving.py 中，修改 _create_chat_logprobs 方法新增 logprob_token_ids 参数，并向下传给 _get_top_logprobs 的 return_all 标志。当 return_all=True 时，_get_top_logprobs 不再根据 top_logprobs 截断结果，而是返回所有条目（含显式 ID）。

5. 批量端点同步：在 vllm/entrypoints/openai/chat_completion/batch_serving.py 中做相似改动，确保批量请求也传递 logprob_token_ids。
6. 测试配套：新增 tests/entrypoints/openai/chat_completion/test_logprob_token_ids.py（208 行），覆盖请求构造函数验证、完整与流式响应对显式 ID 的正确返回、以及错误条件。在 tests/entrypoints/openai/completion/test_completion.py 和 tests/entrypoints/openai/chat_completion/test_batched_chat_completions.py 中添加集成测试。

关键文件：

- vllm/entrypoints/openai/chat_completion/protocol.py（模块 API 协议；类别 source；类型 core-logic；符号 ChatCompletionRequest, to_sampling_params, check_logprobs）：核心变更文件：添加 logprob_token_ids 字段、修改 to_sampling_params 传递逻辑、新增请求验证。
- vllm/entrypoints/openai/completion/protocol.py（模块 API 协议；类别 source；类型 core-logic；符号 CompletionRequest, to_sampling_params, check_logprobs）：并行变更，在 CompletionRequest 中添加相同的字段和逻辑。
- vllm/entrypoints/openai/chat_completion/serving.py（模块 服务层；类别 source；类型 core-logic；符号 _create_chat_logprobs, _get_top_logprobs, chat_completion_stream_generator, chat_completion_full_generator）：服务层关键修改：改造 _create_chat_logprobs 和 _get_top_logprobs 使其支持返回所有显式请求的 ID 对应的 logprobs。
- tests/entrypoints/openai/chat_completion/test_logprob_token_ids.py（模块 测试；类别 test；类型 test-coverage；符号 server, _logprob_entries, _token_id, _top_logprob_token_ids）：新增的专用测试文件，全面覆盖请求构造函数验证、端点集成测试（完整与流式）、错误条件。
- tests/entrypoints/openai/completion/test_completion.py（模块 测试；类别 test；类型 test-coverage；符号 test_logprob_token_ids, test_logprob_token_ids_stream）：在现有补全测试文件中新增两个集成测试，验证显式 ID 在补全端点的完整和流式响应中的正确性。
- tests/entrypoints/openai/chat_completion/test_batched_chat_completions.py（模块 测试；类别 test；类型 test-coverage；符号 test_batched_chat_completions_logprob_token_ids）：在批量聊天完成测试中增加对 logprob_token_ids 的测试，确保批量端点正常工作。

关键符号：ChatCompletionRequest.to_sampling_params, CompletionRequest.to_sampling_params, ChatCompletionRequest.check_logprobs, CompletionRequest.check_logprobs, ChatServing._create_chat_logprobs, ChatServing._get_top_logprobs, ChatServing.chat_completion_stream_generator, ChatServing.chat_completion_full_generator, BatchChatCompletionRequest.check_batch_mode

关键源码片段

vllm/entrypoints/openai/chat_completion/serving.py

服务层关键修改：改造 _create_chat_logprobs 和 _get_top_logprobs 使其支持返回所有显式请求的 ID 对应的 logprobs。

```
def _get_top_logprobs(
    top_logprobs: int | None,
    tokenizer: TokenizerLike | None,
    should_return_as_token_id: bool,
    return_all: bool = False, # 新增：当设置了 logprob_token_ids 时，需返回所有条目
) -> list[ChatCompletionLogProb]:
    """
    将原生 logprobs 字典转为 OpenAI 格式的列表。
    如果 return_all 为 True，则跳过 top_logprobs 截断，确保所有显式请求的 ID 都被包含。
    """
    return [
        ChatCompletionLogProb(
            token=... ,
            logprob=... ,
            bytes=... ,
        )
        for i, p in enumerate(logprobs.items())
        # 显式 ID 模式下无条件包含；否则按 top_logprobs 截断
        if return_all
        or top_logprobs == -1
        or (top_logprobs is not None and i < top_logprobs)
    ]


def _create_chat_logprobs(
    token_ids: list[int],
    top_logprobs: GenericSequence[dict[int, Logprob] | None],
    tokenizer: TokenizerLike | None,
    num_output_top_logprobs: int | None = None,
    logprob_token_ids: list[int] | None = None, # 新增参数
    return_as_token_id: bool | None = None,
) -> ChatCompletionLogProbs:
    """创建 OpenAI 格式的 logprobs。"""
    # ... 原有逻辑 ...
    for i, token_id in enumerate(token_ids):
        # ... 原有处理 ...
        # 在调用 _get_top_logprobs 时传入 return_all
        top_logprob_list = _get_top_logprobs(
            num_output_top_logprobs,
            tokenizer,
            should_return_as_token_id,
            return_all=bool(logprob_token_ids),
        )
        # ...
```

评论区精华

主要讨论集中在 PR 早期包含的 per-request logprobs_mode 部分。njhill 评论指出不应支持每请求 logprobs 模式，因为混合批处理中会引入复杂性和隔离问题。作者接受建议，移除所有与 logprobs_mode 相关的 worker-side 改动和批聚合逻辑，仅保留 logprob_token_ids 的 HTTP 层暴露。AI 审查（gemini-code-assist，depthfirst-app）提出的批隔离和 ValueError 风险也因代码移除而过期，作者在回复中说明新版本已不涉及那些文件。

- 移除 per-request logprobs_mode 支持 (design): 作者采纳建议，移除所有与 logprobs_mode 相关的 worker-side 改动和批聚合逻辑，仅保留 logprob_token_ids 的 HTTP 层暴露。

风险与影响

- 风险：风险较低。核心变更仅限于前端协议层和服务层，不触及采样引擎或模型执行逻辑。新增的验证（beam search 排斥、echo-only 排斥、logprobs 必设）能尽早拒绝非法请求，避免运行时异常。logprob_token_ids 与 top_logprobs 的交互清晰：显式 ID 存在时替代自然 top-k，引擎走已有且验证过的显式收集路径。潜在风险：若用户同时设置 top_logprobs 和 logprob_token_ids，top_logprobs 会被静默忽略（因 logprobs 被设为 None），这可能出乎用户预期，但字段描述已说明优先级。
- 影响：用户：需要为固定小词汇表评分（如多标签分类、PII 检测）的调用方，现在可以通过标准 OpenAI 端点高效获取目标 logprobs，无需绕过底层 API。系统：对性能无负面影响，因为显式 ID 路径原本就存在，且通常请求的 ID 数量远小于 top-k 的默认值。团队：维护成本低，改动集中在前端协议和服务层，未来若需扩展可参照此模式。
- 风险标记：新增请求验证，影响范围仅前端，向下兼容

关联脉络

- 暂无明显关联 PR