

PR #43260 完整报告

vllm-project/vllm

[Frontend] Add truncation side to OpenAI endpoints

合并时间: 2026-05-22 11:32

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/43260>

执行摘要

- 一句话: 为 OpenAI 端点添加截断方向控制
- 推荐动作: 这是一个设计清晰的特性 PR, 改动量小, 覆盖了完整的参数传递与验证链路, 值得前端开发者作为新增 endpoint 参数的参考模板。

功能与动机

当使用 prompt 截断时, 不同场景需要保留 prompt 头部 (如系统指令) 或尾部 (如最新对话)。此参数提供灵活选择, 避免用户手动切片 token 序列。

实现拆解

1. 协议层添加字段: 在 ChatCompletionRequest (vllm/entrypoints/openai/chat_completion/protocol.py) 和 CompletionRequest (vllm/entrypoints/openai/completion/protocol.py) 中新增 truncation_side: Literal["left", "right"] | None 字段, 并传入 build_tok_params 方法。
2. 渲染参数验证与传播: 在 vllm/renderers/params.py 的 TokenizeParams 中, __post_init__ 增加值合法性校验 (仅允许 None、"left"、"right"); with_kwargs 方法从 kwargs 中弹出 truncation_side 并传递给新建的 TokenizeParams。
3. tokenization 核心调整: 在 get_encode_kwargs 中, 当同时指定 truncation_side 和 truncate_prompt_tokens 时, 禁用 tokenizer 原生截断 (truncation=False), 改为由 vLLM 内部函数 _token_truncation 根据方向切片, 避免 tokenizer 默认截断方向干扰。
4. 测试覆盖: 新增 test_serving_chat_truncation_side_controls_prompt_truncation (tests/entrypoints/openai/chat_completion/test_serving_chat.py) 和 test_completion_truncation_side_controls_prompt_truncation (tests/entrypoints/openai/completion/test_completion.py), 分别验证 chat 和 completion 端点下 right/left 截断结果正确。

关键文件:

- vllm/renderers/params.py (模块 渲染器; 类别 source; 类型 core-logic; 符号 post_init, with_kwargs, get_encode_kwargs): 核心渲染参数类, 包含 truncation_side 验证、传播以及 get_encode_kwargs 中禁用 tokenizer 截断的关键逻辑。
- vllm/entrypoints/openai/chat_completion/protocol.py (模块 入口层; 类别 source; 类型 core-logic; 符号 ChatCompletionRequest, build_tok_params):

ChatCompletionRequest 类添加 truncation_side 字段，并在 build_tok_params 中传递至 TokenizeParams。

- vllm/entrypoints/openai/completion/protocol.py (模块 入口层; 类别 source; 类型 core-logic; 符号 CompletionRequest, build_tok_params) : CompletionRequest 类添加 truncation_side 字段，并在 build_tok_params 中传递至 TokenizeParams。
- tests/entrypoints/openai/chat_completion/test_serving_chat.py (模块 对话服务; 类别 test; 类型 test-coverage; 符号 test_serving_chat_truncation_side_controls_prompt_truncation, _render_chat_prompt_token_ids) : 新增测试函数 test_serving_chat_truncation_side_controls_prompt_truncation 及辅助函数 _render_chat_prompt_token_ids, 覆盖 chat 端点两种截断方向。
- tests/entrypoints/openai/completion/test_completion.py (模块 补全服务; 类别 test; 类型 test-coverage; 符号 test_completion_truncation_side_controls_prompt_truncation) : 新增测试函数 test_completion_truncation_side_controls_prompt_truncation, 验证 completion 端点两种截断方向。

关键符号: test_serving_chat_truncation_side_controls_prompt_truncation, _render_chat_prompt_token_ids, test_completion_truncation_side_controls_prompt_truncation, post_init, with_kwargs, get_encode_kwargs, build_tok_params

关键源码片段

vllm/renderers/params.py

核心渲染参数类，包含 truncation_side 验证、传播以及 get_encode_kwargs 中禁用 tokenizer 截断的关键逻辑。

```
def get_encode_kwargs(self) -> dict[str, Any]:
    """The arguments to pass to `tokenizer.encode`."""
    # Allow one extra token so truncation looks correct
    # while still failing `self._token_len_check`.
    max_length = self.max_input_tokens + 1

    # Explicit truncation-side overrides require the full token sequence
    # so we can slice from the requested side in `_token_truncation`.
    # Disable tokenizer-level truncation because generation tokenizers
    # default to left truncation while callers may request right truncation.
    if self.truncation_side is not None and self.truncate_prompt_tokens is not None:
        return dict(
            truncation=False, # disable tokenizer truncation
            add_special_tokens=self.add_special_tokens,
            max_length=None,
            padding=False,
        )

    # Otherwise, allow tokenizer truncation up to max_length
    # or explicit truncate_prompt_tokens if set.
    return dict(
        truncation=max_length if self.truncate_prompt_tokens is None
```

```

        else self.truncate_prompt_tokens,
        add_special_tokens=self.add_special_tokens,
        max_length=max_length,
        padding=False,
    )

```

vllm/entrypoints/openai/chat_completion/protocol.py

ChatCompletionRequest 类添加 truncation_side 字段，并在 build_tok_params 中传递至 TokenizeParams。

```

class ChatCompletionRequest(OpenAIBaseModel):
    # ... existing fields ...
    truncation_side: Literal["left", "right"] | None = Field(
        default=None,
        description=(
            "Which side to truncate from when truncate_prompt_tokens is active. "
            "'right' keeps the first N tokens. "
            "'left' keeps the last N tokens."
        ),
    )
    # ... other fields ...

def build_tok_params(self, model_config: ModelConfig) -> TokenizeParams:
    # ... existing logic ...
    return TokenizeParams(
        max_total_tokens=model_config.max_model_len,
        max_output_tokens=max_output_tokens or 0,
        truncate_prompt_tokens=self.truncate_prompt_tokens,
        truncation_side=self.truncation_side, # newly passed
        add_special_tokens=self.add_special_tokens,
        needs_detokenization=bool(self.echo and not self.return_token_ids),
        max_total_tokens_param="max_model_len",
        max_output_tokens_param=max_output_tokens_param,
    )

```

评论区精华

Reviewer DarkLight1337 指出作者最初对 [vllm/inputs/preprocess.py](#) 的改动应避免，因为该代码路径已废弃，并将于 PR #38688 中移除。作者接受建议并回退了该文件变更。其余评论无实质讨论。

- 预处理器废弃路径变更 (design): 作者接受建议并回退了该文件变更。

风险与影响

- 风险：主要风险在于 tokenization 逻辑调整：当显式指定 truncation_side 时，禁用 tokenizer 级截断可能与其他参数（如 add_special_tokens）产生细微交互。另外，tokenizers 库本身的默认截断方向可能是左或右，此改动统一交由 vLLM 处理，在 edge case 下可能产生与之前不同的 token 序列，但仅当用户主动设置 truncation_side 时生效

，故向后兼容。

- 影响：用户影响：新增可选参数，默认行为不变，用户通过设置 `truncation_side` 获得截断方向控制。系统影响：无性能、安全风险。团队影响：无。
- 风险标记：参数向后兼容，tokenizer 截断方向覆盖

关联脉络

- 暂无明显关联 PR