PR #45876 完整报告

vllm-project/vllm

[Rust Frontend] Validate tokenized bad_words vocabulary range

合并时间：2026-06-18 10:28

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/45876

PR #45876 分析报告：Rust 前端 bad_words 词表越界校验

执行摘要

本 PR 为 Rust 采样参数降级路径补充了 bad_words 词表越界校验，使得 tokenize 后的 bad words token IDs 在进入 engine-core 前被拦截，与 Python 侧行为一致。变更范围小（2 个文件，61 行新增），无风险，已由 BugenZhao 审批通过。

功能与动机

Rust 降级路径会将 bad_words 字符串通过 tokenizer tokenize 得到 _bad_words_token_ids，但此前并未对这些 token ID 进行词表范围校验。Python 侧在 SamplingParams.update_from_tokenizer() 中会校验并抛出 bad_words 校验错误。缺少此检查时，越界的 bad words token ID 会传递到 sampler / logits masking 路径，引发较模糊的 engine 侧错误。PR 旨在填补这一校验空白，提升错误明确性和跨语言一致性。

实现拆解

1. 在 rust/src/text/src/lower/token_ids.rs 中新增校验分支 - 在 validate_vocab_range() 函数中，添加对 params.bad_words_token_ids 的 validate_param 调用，参数名设为 "bad_words"，校验边界为 limits.tokenizer_vocab_size。 - 使用 .iter().flatten().copied() 将嵌套的 Vec<Vec<u32>> 展平为单个 token ID 迭代器（因为每组 bad words 可能对应多个 token ID）。
2. 在 rust/src/text/src/lower.rs 中新增测试桩 FixedTokenizer - 为支持测试中生成非空、可预测的 token ID 列表，新增实现 Tokenizer trait 的 FixedTokenizer，其 encode 方法直接返回预设的 token ID 列表，decode 和 token_to_id 保持桩行为。
3. 新增测试 lower_sampling_params_rejects_out_of_vocab_bad_words - 使用 FixedTokenizer 将字符串 "blocked" tokenize 为 [1999, 2000]，其中 2000 超出测试设置的 tokenizer_vocab_size=2000（合法范围为 0 到 tokenizer_vocab_size-1）。 - 断言返回 Error::OutOfVocab(OutOfVocabError { parameter: "bad_words", token_ids: vec![2000], vocab_size: 2000 })。

rust/src/text/src/lower/token_ids.rs

核心校验逻辑所在，新增 bad_words_token_ids 边界检查分支。

```
// rust/src/text/src/lower/token_ids.rs
```

```
pub(crate) fn validate_vocab_range(
    params: &SamplingParams,
    limits: &SamplingLimits,
) -> Result<(), OutOfVocabError> {
    // 已有校验： stop_token_ids、allowed_token_ids、logit_bias、logprob_token_ids
    // ...

    // 新增：校验 bad_words_token_ids 中每个 token ID 是否 < tokenizer_vocab_size
    if let Some(bad_words_token_ids) = params.bad_words_token_ids.as_deref() {
        validate_param(
            "bad_words", // 参数名，用于错误消息
            bad_words_token_ids.iter().flatten().copied(), // 展开嵌套 Vec<Vec<u32>> 为平坦迭代器
            limits.tokenizer_vocab_size, // 使用 tokenizer 词表大小作为边界
        )?;
    }

    Ok(())
}
```

rust/src/text/src/lower.rs

测试 StubTokenizer 扩展为 FixedTokenizer，并新增坏词越界测试用例。

```
// rust/src/text/src/lower.rs 测试模块

// 新增测试桩：返回固定 token ID 列表的 tokenizer
struct FixedTokenizer {
    token_ids: Vec<u32>,
}

impl Tokenizer for FixedTokenizer {
    fn encode(&self, _text: &str, _add_special_tokens: bool) -> vllm_tokenizer::Result<Vec<u32>>
    {
        Ok(self.token_ids.clone()) // 直接返回预设的 token IDs，模拟 tokenize 结果
    }
    fn decode(&self, _token_ids: &[u32], _skip_special_tokens: bool) -> vllm_tokenizer::
    Result<String> {
        Ok(String::new())
    }
    fn token_to_id(&self, _token: &str) -> Option<u32> {
        None
    }
}

#[test]
fn lower_sampling_params_rejects_out_of_vocab_bad_words() {
    // 模拟将 "blocked" tokenize 成 [1999, 2000]，其中 2000 超出 tokenizer_vocab_size=
    2000（合法范围：0..2000）
    let tokenizer = FixedTokenizer {
        token_ids: vec![1999, 2000],
```

```
    };
    let error = lower_sampling_params(
        SamplingParams {
            bad_words: Some(vec!["blocked".to_string()]),
            ..Default::default()
        },
        SamplingHints::default(),
        sample_sampling_limits(), // tokenizer_vocab_size = 2000
        3,
        &tokenizer,
    )
    .unwrap_err();

    // 断言返回 OutOfVocab 错误，参数名 "bad_words"，越界 token ID 为 [2000]
    assert!(matches!(
        error,
        Error::OutOfVocab(OutOfVocabError {
            parameter: "bad_words",
            token_ids,
            vocab_size: 2000,
        }) if token_ids == vec![2000]
    ));
}
```

评论区精华

> BugenZhao: “The only source of bad_words_token_ids in the current implementation is tokenize_bad_words, so if there are any out-of-vocabulary tokens, it must be a bug in the tokenizer itself. But perhaps it is still useful to have this.”

作者接受了这一观点，校验作为额外安全网保留。

风险与影响

- 风险：低。变更范围小（2 个文件，61 行新增），仅增加校验逻辑，无回归风险。FixedTokenizer 仅用于测试，不影响生产代码。
- 影响：越界的 bad words token IDs 将以清晰的 OutOfVocabError 被提前拒绝，而非引擎侧模糊失败。与 Python 侧行为对齐，降低跨语言行为差异排查成本。无性能影响。

关联脉络

本 PR 是 Rust 前端采样参数校验系列的一部分，此前已为 stop_token_ids、allowed_token_ids、logit_bias、logprob_token_ids 等参数添加了类似校验（参见 token_ids.rs 中已有的 validate_vocab_range 函数）。本次补齐了 bad_words 的校验，实现了全面覆盖。