PR #46781 完整报告

vllm-project/vllm

[Model Runner V2][Spec Decode] Implement block verification for rejection sampling

合并时间：2026-06-30 23:07

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/46781

执行摘要

- 一句话：实现块验证拒绝采样，提升推测解码接受率
- 推荐动作：该 PR 值得精读，特别是对推测解码和 Triton 内核开发感兴趣的工程师。实现展示了如何基于论文在推理引擎中集成复杂采样算法，并附带了完整的基准测试和测试验证。

功能与动机

当前 Model Runner V2 不支持块验证拒绝采样，但 V1 正在支持（#40819）。块验证被期望产生至少与标准拒绝采样一样高的接受率，且成本相当。实现此功能可以为推测解码场景带来更高的效率和吞吐量。

实现拆解

1. 配置扩展：在 vllm/config/speculative.py 中，RejectionSampleMethod 类型别名新增 block 字面量，并在文档字符串中说明。
2. 采样器集成：在 vllm/v1/worker/gpu/spec_decode/rejection_sampler.py 的 RejectionSampler.__init__ 中根据配置设置 self.use_block_verification，并在 __call__ 中传递给 rejection_sample。
3. 核心 Triton 内核：在 vllm/v1/worker/gpu/spec_decode/rejection_sampler_utils.py 中重命名辅助函数，新增 _compute_global_residual_mass、_compute_global_target_argmax、_compute_global_logprobs_and_logsumexp、_compute_cumulative_log_p_kernel、_compute_local_residual_mass_kernel 等内核，并重写 _compute_block_stats_kernel 以支持控制流分支。
4. 测试覆盖：在 tests/v1/spec_decode/test_rejection_sampler_utils.py 中添加两个测试，验证分布正确性和接受长度担保。

关键文件：

- vllm/v1/worker/gpu/spec_decode/rejection_sampler_utils.py（模块 推测解码；类别 source；类型 core-logic；符号 _compute_block_max_and_sumexp, _compute_max_and_sumexp, _compute_global_lse, _compute_global_logsumexp）：核心实现文件，包含所有块验证相关的 Triton 内核和算法逻辑。
- vllm/v1/worker/gpu/spec_decode/rejection_sampler.py（模块 推测解码；类别 source；类型 core-logic；符号 RejectionSampler.init, RejectionSampler.call）：采样器类，将配置标志传递到核心采样函数。

- vllm/config/speculative.py（模块 配置层；类别 source；类型 core-logic；符号 RejectionSampleMethod）：配置变更，新增 block 为 rejection_sample_method 的可选值。
- tests/v1/spec_decode/test_rejection_sampler_utils.py（模块 测试覆盖；类别 test；类型 test-coverage；符号 test_block_verification_rejection_sample, test_block_verification_accepts_at_least_as_many）：新增测试用例验证块验证的正确性和接受率保证。

关键符号：_compute_block_max_and_sumexp, _compute_max_and_sumexp, _compute_global_lse, _compute_global_logsumexp, _compute_block_stats_kernel, _compute_global_residual_mass, _compute_global_target_argmax, _compute_global_logprobs_and_logsumexp, _compute_cumulative_log_p_kernel, _compute_local_residual_mass_kernel, test_block_verification_rejection_sample, test_block_verification_accepts_at_least_as_many, RejectionSampler.init, RejectionSampler.call

关键源码片段

vllm/v1/worker/gpu/spec_decode/rejection_sampler_utils.py

核心实现文件，包含所有块验证相关的 Triton 内核和算法逻辑。

下面是块验证核心内核 _compute_global_residual_mass 的实现。该内核计算残差归一化因子 Z_i，用于接受阈值和重采样分布。

```
@triton.jit
def _compute_global_residual_mass(
    local_residual_mass_ptr,
    local_residual_mass_stride,
    prefix_joint_ratio,
    target_logits_ptr,
    target_logits_stride,
    target_local_max_ptr,
    target_local_max_stride,
    target_local_sumexp_ptr,
    target_local_sumexp_stride,
    draft_sampled_ptr,
    logit_idx,
    vocab_num_blocks,
    PADDED_VOCAB_NUM_BLOCKS: tl.constexpr,
    HAS_DRAFT_LOGITS: tl.constexpr,
):
    if HAS_DRAFT_LOGITS:
        # 完整 draft 分布：累加各子块残差质量
        blocks = tl.arange(0, PADDED_VOCAB_NUM_BLOCKS)
        mask = blocks < vocab_num_blocks
        partials = tl.load(
            local_residual_mass_ptr + logit_idx * local_residual_mass_stride + blocks,
            mask=mask,
```

```
            other=0.0,
        )
        return tl.sum(partials, axis=0)
    else:
        # one-hot（贪婪）draft：M_s 是 draft_token 上的点质量
        # 残差质量 = prefix_joint_ratio * (1 - M_b(draft_token))
        draft_token = tl.load(draft_sampled_ptr + logit_idx + 1).to(tl.int64)
        target_lse = _compute_global_logsumexp(
            target_local_max_ptr,
            target_local_max_stride,
            target_local_sumexp_ptr,
            target_local_sumexp_stride,
            logit_idx,
            vocab_num_blocks,
            PADDED_VOCAB_NUM_BLOCKS,
        )
        target_logit = tl.load(
            target_logits_ptr + logit_idx * target_logits_stride + draft_token,
        ).to(tl.float32)
        m_b = tl.exp(target_logit - target_lse)
        return prefix_joint_ratio * (1.0 - m_b)
```

vllm/v1/worker/gpu/spec_decode/rejection_sampler.py

采样器类，将配置标志传递到核心采样函数。

下面是采样器类的接口变更，展示了 use_block_verification 标志的设置和传递。

```
class RejectionSampler:
    def __init__(
        self,
        sampler: Sampler,
        spec_config: SpeculativeConfig,
        device: torch.device,
    ):
        self.sampler = sampler
        self.num_speculative_steps = spec_config.num_speculative_tokens
        rejection_sample_method = spec_config.rejection_sample_method
        self.use_block_verification: bool = False
        self.synthetic_conditional_rates: torch.Tensor | None = None
        if rejection_sample_method == "synthetic":
            # 合成拒绝采样：使用预设的接受率
            assert spec_config.synthetic_acceptance_rates is not None
            self.synthetic_conditional_rates = torch.tensor(
                unconditional_to_conditional_rates(
                    spec_config.synthetic_acceptance_rates
                ),
                dtype=torch.float32,
                device=device,
            )
        elif rejection_sample_method == "block":
```

```
            # 启用块验证
            self.use_block_verification = True

    def __call__(
        self,
        logits: torch.Tensor,
        input_batch: InputBatch,
        draft_logits: torch.Tensor | None = None,
    ) -> SamplerOutput:
        ...
        sampled, num_sampled = rejection_sample(
            processed_logits,
            draft_logits,
            draft_sampled,
            input_batch.cu_num_logits,
            pos,
            input_batch.idx_mapping,
            input_batch.expanded_idx_mapping,
            input_batch.expanded_local_pos,
            self.sampler.sampling_states.temperature.gpu,
            self.sampler.sampling_states.seeds.gpu,
            self.num_speculative_steps,
            self.synthetic_conditional_rates,
            use_fp64=self.sampler.use_fp64_gumbel,
            use_block_verification=self.use_block_verification, # 传递块验证标志
        )
        ...
```

评论区精华

该 PR 没有引发实质性的技术讨论。WoosukKwon 在审查后批准了 PR，表示 'LGTM. Thanks for doing this!'。此前 mergify[bot] 提示存在合并冲突（后已解决）。

- 暂无高价值评论线程

风险与影响

- 风险：主要风险集中在 vllm/v1/worker/gpu/spec_decode/rejection_sampler_utils.py 中的新 Triton 内核。数值计算（指数、对数）可能引入精度问题，尤其是在低精度模式下。块验证算法新增控制流路径，可能影响标准模式和合成模式的正确性。需要确保新内核与 GPU 架构的兼容性（已在 CUDA 上测试）。
- 影响：对用户：提供 block 作为 rejection_sample_method 的新值，用户可通过配置启用。对系统：新增一个或两个额外的 Triton 内核启动，但计算量相对小，且可能减少拒绝步骤，从而提升总体吞吐量。对开发团队：需要维护额外的采样方法，文档和测试已更新。
- 风险标记：核心路径变更，数值精度，新增 API 配置

关联脉络

- PR #40819 [V1] Implement block verification for rejection sampling: V1 中相同的块验证功能正在支持，本 PR 是 V2 的实现，两者算法相同但代码路径不同。