PR #40662 完整报告

vllm-project/vllm

[Feat] Unified Synthetic Acceptance Rate for V1 and V2

合并时间：2026-04-24 08:48

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/40662

执行摘要

- 一句话：统一 V1 和 V2 合成拒绝采样接受率配置
- 推荐动作：建议仔细阅读 vllm/config/speculative.py 中的配置解析和校验逻辑（最小方差调度设计简洁），以及 Triton kernel 的改动。对于自行实现推测解码的开发者，unconditional_to_conditional_rates 转换函数值得复用。PR 整体设计合理，测试覆盖完整，应批准合并。

功能与动机

原合成拒绝采样仅支持 V2 且使用固定的几何衰减模型，无法精细控制每个位置的接受率。本 PR 将其扩展到 V1 模型运行器，并允许用户通过逐位置接受率列表或期望的平均接受长度来精确控制推测解码行为。PR body 提到“expands support of synthetic acceptance rate to ModelRunnerV1”和“adds support for per-position synthetic acceptance rate selection”。

实现拆解

- 步骤 1：配置层革新（vllm/config/speculative.py） 将旧字段 synthetic_acceptance_rate 拆分为两个互斥字段 synthetic_acceptance_rates（逐位置无条件接受率列表）和 synthetic_acceptance_length（目标平均接受长度）。新增静态方法 _acceptance_length_to_rates 将平均长度转换为最小方差调度下的无条件接受率序列，以及 _resolve_synthetic_acceptance_rates 统一校验并解析两个参数，确保恰好提供一个且值合法。
- 步骤 2：Triton 内核改造（vllm/v1/worker/gpu/spec_decode/synthetic_rejection_sampler_utils.py） 重写 _synthetic_rejection_sample_kernel，将原来的 base_acceptance_rate 和 decay_factor 标量参数替换为一个长度为 num_speculative_steps 的接受率张量指针 acceptance_rates_ptr。循环内改为从该张量按位置加载接受率，去掉了 decay_factor 的累乘逻辑。同时删除了已无用的 compute_synthetic_rejection_sampler_params、mean_joint_prob 等辅助函数。
- 步骤 3：采样器统一（vllm/v1/sample/rejection_sampler.py 和 vllm/v1/worker/gpu/spec_decode/rejection_sampler.py） V1 的 RejectionSampler 和 V2 的 RejectionSampler 在 __init__ 中接收 SpeculativeConfig，当 rejection_sample_method 为 'synthetic' 时，通过 unconditional_to_conditional_rates 将无条件接受率转换为条件接受率并缓存为 GPU 张量。forward 方法（V1）和 __call__（V2）将条件率传入底层的采样内核。

- 步骤 4：条件率转换工具（vllm/v1/spec_decode/utils.py） 新增 unconditional_to_conditional_rates 函数，实现 c_0 = p_0，c_i = p_i / p_{i-1} 的转换，并对零值进行截断处理，确保后续位置的条件率为 0。
- 步骤 5：测试配套 更新 tests/v1/spec_decode/test_synthetic_rejection_sampler_utils.py：新增对 unconditional_to_conditional_rates 的基础值、零值、全一的测试，以及对 _acceptance_length_to_rates 和 _resolve_synthetic_acceptance_rates 的参数化测试。新增 tests/v1/sample/test_rejection_sampler.py 中的 test_synthetic_all_accepted 和 test_synthetic_all_rejected 用例，验证 V1 合成模式在贪心和非贪心下的全接受 / 全拒绝场景。 补充 tests/v1/e2e/spec_decode/test_spec_decode.py 中的端到端测试 test_synthetic_acceptance_rate，验证命令行参数与日志输出的一致性。

关键文件：

- vllm/config/speculative.py（模块 配置层；类别 source；类型 core-logic；符号 _acceptance_length_to_rates, _resolve_synthetic_acceptance_rates）：配置入口，定义逐位置接受率列表和目标平均长度字段及解析方法，是统一配置的核心。
- vllm/v1/worker/gpu/spec_decode/synthetic_rejection_sampler_utils.py（模块 推测解码；类别 source；类型 core-logic；符号 compute_synthetic_rejection_sampler_params, mean_joint_prob, min_valid_decay_factor, compute_base_acceptance_rate）：Triton kernel 核心修改，从几何衰减改为逐位置接受率张量。
- vllm/v1/sample/rejection_sampler.py（模块 采样器；类别 source；类型 core-logic；符号 init）：V1 采样器集成合成拒绝采样模式，初始化条件率张量。
- vllm/v1/worker/gpu/spec_decode/rejection_sampler.py（模块 采样器；类别 source；类型 dependency-wiring）：V2 采样器适配新配置接口，传递条件率张量。
- vllm/v1/spec_decode/utils.py（模块 工具函数；类别 source；类型 core-logic；符号 unconditional_to_conditional_rates）：新增无条件转条件接受率函数，被两个采样器共同调用。
- tests/v1/spec_decode/test_synthetic_rejection_sampler_utils.py（模块 单元测试；类别 test；类型 test-coverage；符号 test_unconditional_to_conditional_rates_basic, test_unconditional_to_conditional_rates_handles_zero, test_unconditional_to_conditional_rates_all_ones, test_acceptance_length_to_rates）：覆盖新核心函数（无条件转条件率、长度转率、配置解析）。
- tests/v1/sample/test_rejection_sampler.py（模块 集成测试；类别 test；类型 test-coverage；符号 _make_synthetic_sampler, _make_sampling_metadata, test_synthetic_all_accepted, test_synthetic_all_rejected）：验证 V1 合成模式整体正确性。
- tests/v1/e2e/spec_decode/test_spec_decode.py（模块 端到端测试；类别 test；类型 test-coverage；符号 test_synthetic_acceptance_rate）：端到端测试验证命令行配置与日志输出。

关键符号：unconditional_to_conditional_rates, _acceptance_length_to_rates, _resolve_synthetic_acceptance_rates, RejectionSampler.init, synthetic_rejection_sample, compute_synthetic_rejection_sampler_params, mean_joint_prob, min_valid_decay_factor, compute_base_acceptance_rate

关键源码片段

vllm/config/speculative.py

配置入口，定义逐位置接受率列表和目标平均长度字段及解析方法，是统一配置的核心。

```
# vllm/config/speculative.py (head 版本 )
# 新增字段和静态方法，用于解析合成拒绝采样的逐位置接受率

# 新配置字段：互斥的合成接受率列表或目标平均长度
synthetic_acceptance_rates: list[float] | None = None
synthetic_acceptance_length: float | None = None

@staticmethod
def _acceptance_length_to_rates(length: float, n: int) -> list[float]:
    """Mean acceptance length to unconditional per-position rates, using
    the minimum-variance schedule."""
    # 输入：期望平均接受长度（1 到 n+1），输出：长度为 n 的无条件接受率列表
    num_drafts = length - 1 # 期望接受的 draft token 数
    num_full = int(num_drafts)
    # 构造最小方差调度：前 num_full 个位置为 1.0，中间一个为小数部分，其余为 0.0
    return (
        [1.0] * num_full + [num_drafts - num_full] + [0.0] * (n - num_full - 1)
    )[:n]

@staticmethod
def _resolve_synthetic_acceptance_rates(
    n: int,
    rates: list[float] | None,
    length: float | None,
) -> list[float]:
    """Return per-position unconditional acceptance rates from exactly one
    of `rates` or `length` (validates range, length, and monotonicity)."""
    # 确保恰好提供一个参数
    if (rates is None) == (length is None):
        raise ValueError(
            "rejection_sample_method='synthetic' requires exactly one of "
            "synthetic_acceptance_rates or synthetic_acceptance_length."
        )
    if rates is not None:
        # 校验长度、范围、单调非增
        if len(rates) != n:
            raise ValueError(
                f"synthetic_acceptance_rates must have length {n}, got {rates}."
            )
        if not all(0.0 <= r <= 1.0 for r in rates):
            raise ValueError(
                f"synthetic_acceptance_rates entries must be in [0, 1], got {rates}."
            )
        if any(rates[i] > rates[i - 1] for i in range(1, n)):
```

```
            raise ValueError(
                f"synthetic_acceptance_rates must be non-increasing, got {rates}."
            )
        return list(rates)
    # 使用长度参数，校验范围后转换
    assert length is not None
    if not 1.0 <= length <= float(n + 1):
        raise ValueError(
            f"synthetic_acceptance_length must be in [1, {n + 1}], got {length}."
        )
    return SpeculativeConfig._acceptance_length_to_rates(length, n)
```

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

Triton kernel 核心修改，从几何衰减改为逐位置接受率张量。

```
# vllm/v1/worker/gpu/spec_decode/synthetic_rejection_sampler_utils.py (head 版本 )
# Triton 内核及封装函数，使用逐位置接受率张量

import torch
from vllm.triton_utils import tl, triton
from vllm.v1.worker.gpu.sample.gumbel import tl_rand64


@triton.jit
def _synthetic_rejection_sample_kernel(
    sampled_ptr, sampled_stride, # [num_reqs, num_speculative_steps + 1] 输出缓冲
    num_sampled_ptr, # [num_reqs] 每个请求实际采样的 token 数
    target_sampled_ptr, # [num_draft_tokens + num_reqs] 目标模型采样结果
    input_ids_ptr, # [num_draft_tokens + num_reqs] draft token ID
    cu_num_logits_ptr, # [num_reqs + 1] 每个请求的 logits 偏移
    pos_ptr, # [num_logits] 位置编码
    idx_mapping_ptr, # [num_reqs] 请求索引映射
    seeds_ptr, # [max_num_reqs] 随机种子
    # 新参数：逐位置接受率张量，长度等于 num_speculative_steps
    acceptance_rates_ptr, # [num_speculative_steps]
):
    req_idx = tl.program_id(0)
    start_idx = tl.load(cu_num_logits_ptr + req_idx)
    end_idx = tl.load(cu_num_logits_ptr + req_idx + 1)
    num_tokens = end_idx - start_idx
    req_state_idx = tl.load(idx_mapping_ptr + req_idx)
    seed = tl.load(seeds_ptr + req_state_idx)

    num_sampled = 0
    rejected = False
    for i in range(num_tokens - 1):
        if not rejected:
            logit_idx = start_idx + i
            pos = tl.load(pos_ptr + logit_idx)
```

```
            # 生成均匀随机数用于接受判断
            u = tl_rand64(seed, pos, includes_zero=False)
            # 从张量中直接读取该位置的条件接受率（已由前处理转换为条件概率）
            acceptance_rate = tl.load(acceptance_rates_ptr + i)
            if u < acceptance_rate:
                # 接受 draft token
                sampled = tl.load(input_ids_ptr + logit_idx + 1).to(tl.int64)
            else:
                # 拒绝，采用目标采样值
                sampled = tl.load(target_sampled_ptr + logit_idx)
                rejected = True
            tl.store(sampled_ptr + req_idx * sampled_stride + i, sampled)
            num_sampled += 1
    if not rejected:
        # 所有 draft 均接受，取 bonus token（目标模型最后一个采样）
        target_sampled = tl.load(target_sampled_ptr + start_idx + num_tokens - 1)
        tl.store(sampled_ptr + req_idx * sampled_stride + num_tokens - 1, target_sampled)
        num_sampled += 1
    tl.store(num_sampled_ptr + req_idx, num_sampled)


def synthetic_rejection_sample(
    target_sampled: torch.Tensor,
    draft_sampled: torch.Tensor,
    cu_num_logits: torch.Tensor,
    pos: torch.Tensor,
    idx_mapping: torch.Tensor,
    seed: torch.Tensor,
    # 新参数：预计算的条件接受率张量，shape [num_speculative_steps]
    acceptance_rates: torch.Tensor,
    num_speculative_steps: int,
) -> tuple[torch.Tensor, torch.Tensor]:
    num_reqs = cu_num_logits.shape[0] - 1
    sampled = target_sampled.new_empty(num_reqs, num_speculative_steps + 1)
    num_sampled = target_sampled.new_empty(num_reqs, dtype=torch.int32)
    # 启动 Triton kernel，传入接受率张量指针
    _synthetic_rejection_sample_kernel[(num_reqs,)](
        sampled, sampled.stride(0), num_sampled,
        target_sampled, draft_sampled, cu_num_logits, pos, idx_mapping, seed,
        acceptance_rates,
        num_warps=1,
    )
    return sampled, num_sampled
```

vllm/v1/sample/rejection_sampler.py

V1 采样器集成合成拒绝采样模式，初始化条件率张量。

```
# vllm/v1/sample/rejection_sampler.py (head 版本 )
# RejectionSampler 构造函数，增加合成拒绝采样支持
```

```
from __future__ import annotations
from typing import TYPE_CHECKING
from vllm.v1.spec_decode.utils import unconditional_to_conditional_rates

if TYPE_CHECKING:
    from vllm.config.speculative import SpeculativeConfig

class RejectionSampler(nn.Module):

    def __init__(
        self,
        sampler: Sampler,
        # 新增可选参数：推测配置和设备，用于初始化合成模式的条件率
        spec_config: SpeculativeConfig | None = None,
        device: torch.device | None = None,
    ):
        super().__init__()
        self.sampler = sampler
        logprobs_mode = self.sampler.logprobs_mode
        self.is_processed_logprobs_mode = logprobs_mode.startswith("processed")
        self.is_logits_logprobs_mode = logprobs_mode.endswith("logits")

        # 合成拒绝采样的条件接受率张量（GPU），初始化为 None
        self.synthetic_conditional_rates: torch.Tensor | None = None
        if (
            spec_config is not None
            and spec_config.rejection_sample_method == "synthetic"
        ):
            # 确保配置已解析出自洽的逐位置接受率列表
            assert spec_config.synthetic_acceptance_rates is not None
            # 将无条件接受率转换为条件接受率，并搬到 GPU
            self.synthetic_conditional_rates = torch.tensor(
                unconditional_to_conditional_rates(
                    spec_config.synthetic_acceptance_rates
                ),
                dtype=torch.float32,
                device=device,
            )
        # 标记当前是否处于合成模式（条件率不为 None）
        self.synthetic_mode = self.synthetic_conditional_rates is not None
```

评论区精华

Review 中 gemini-code-assist[bot] 提出了 Triton kernel 中 token ID 类型不一致的问题（int64 与 int32 混用），但作者 benchislett 认为该问题属于幻觉且超出本 PR 范围，仅在非必要位置清理了类型转换。TheEpicDolphin 建议保留原注释，作者认为注释冗余未采纳。最终 TheEpicDolphin 和 WoosukKwon 均批准合并。

- Triton kernel 中 token ID 类型不一致问题 (correctness): 作者 benchislett 认为该问题超出本 PR 范围且不影响运行（out of scope），仅在非必要位置清理了类型转换。未进行全局修复。
- 注释保留建议 (style): 作者 benchislett 认为代码本身已有 if accepted 和 else rejected = True，注释冗余，未采纳。

风险与影响

- 风险：
  - 类型兼容性风险：Triton kernel 中 draft token 使用了 int64 而目标采样和输出缓冲为 int32，可能导致隐式类型转换问题。作者认为不影响运行但未全面修复。
  - 配置迁移风险：旧字段 synthetic_acceptance_rate 被移除，使用该字段的已有配置文件会报错，但 PR 会提供明确的错误提示。
  - 性能风险：Triton kernel 新增了一次 tl.load（加载接受率），但该操作在循环内开销很小，不会造成显著性能退化。
- 影响：
  - 用户：V1 用户现在可以使用合成拒绝采样；配置方式更灵活，可由逐位置列表或目标平均长度指定。旧配置需迁移。
  - 系统：统一了 V1/V2 的合成拒绝采样路径，简化了未来维护。V2 的内核调用签名改变，需同步更新其他调用方（已在本 PR 中完成）。
  - 团队：代码量减少（-137 行），逻辑更清晰，新配置校验严格，减少误用。
  - 风险标记：类型兼容性风险，配置迁移风险，Triton kernel 改动影响面

关联脉络

- 暂无明显关联 PR