PR #49531 完整报告

vllm-project/vllm

[Perf] DeepSeek-OCR-2 TTFT Optimize

合并时间：2026-07-26 13:53

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/49531

执行摘要

- 一句话：优化 DeepSeek-OCR-2 注意力掩码计算，TTFT 降 46%
- 推荐动作：值得精读。该 PR 清晰展示了如何通过分析数据模式（批次不变性）消除冗余 CPU→GPU 操作，并利用 @lru_cache 以极小成本实现加速。设计决策中关于缓存 key 范围的取舍（保留 dtype 和 device）体现了对实际部署场景的考量。

功能与动机

create_custom_4d_mask 函数在 CPU 上循环构造掩码，是 TTFT 的瓶颈。PR body 中指明该函数是 CPU-bound，torch profiler 显示 aten::index_put_impl 调用达 229,296 次，消耗 52.2% 的 CPU 时间。优化目标是将这一计算移至 GPU 并消除冗余索引操作。

实现拆解

1. 移除实例状态：删除 CustomQwen2ModelInner.forward 中的 _current_token_type_ids 实例变量，不再传递 token_type_ids，因为 mask 生成不再依赖每个样本的 token_type_ids。
2. 新增类方法 compute_mask_base：在 CustomQwen2ModelInner 中声明类方法，使用 @classmethod 和 @lru_cache(maxsize=8) 缓存。该方法根据固定模式（前一半 image token、后一半 text token）在 GPU 上一次性计算 [1,1,S,S] 掩码基座。
3. 简化 _create_custom_4d_mask：原方法遍历 batch、逐位置填充 mask。改为调用 compute_mask_base 获得基座，然后通过 expand(batch_size,-1,-1,-1) 广播到 batch 维度。
4. 调整调用链：_update_causal_mask 不再传递 token_type_ids，直接调用 _create_custom_4d_mask 并处理 padding mask。
5. 性能验证：未添加单元测试，但 PR 中附带了详细的 TTFT 基准脚本和分析，证明优化效果。

关键文件：

- vllm/model_executor/models/deepencoder2.py（模块 视觉编码；类别 source；类型 core-logic；符号 CustomQwen2Decoder, compute_mask_base, _create_custom_4d_mask, _update_causal_mask）：核心性能优化文件，重构了掩码计算和缓存逻辑

关键符号：compute_mask_base, _create_custom_4d_mask, _update_causal_mask

关键源码片段

vllm/model_executor/models/deepencoder2.py

核心性能优化文件，重构了掩码计算和缓存逻辑

```
from functools import lru_cache
import torch

# ... 在 CustomQwen2ModelInner 类中 ...

@classmethod
@lru_cache(maxsize=8)
def compute_mask_base(cls, sequence_length: int, dtype: torch.dtype, device: torch.device):
    """
    Compute the base 4D attention mask for DeepSeek-OCR-2.
    The mask is batch-invariant because token_type_ids follows a fixed pattern:
    first half of tokens are image tokens (non-causal), second half are text tokens (causal).
    This method computes a single [1, 1, S, S] mask and relies on expand()
    in the caller to broadcast over batch dimension.
    """
    min_dtype = torch.finfo(dtype).min
    n_query = sequence_length // 2 # image token count (also text token count)
    # Indices of image tokens (first half)
    img = torch.arange(sequence_length, device=device) < n_query
    txt = ~img
    # Standard causal mask: lower triangular
    causal = torch.tril(torch.ones(sequence_length, sequence_length,
                                   dtype=torch.bool, device=device))
    # Allow: image token attends to all; text token attends to all image tokens
    # plus causal among text tokens.
    allow = img[None, :] | (txt[:, None] & txt[None, :] & causal)
    return torch.where(
        allow,
        torch.zeros((), dtype=dtype, device=device),
        torch.full((), min_dtype, dtype=dtype, device=device),
    )[None, None] # add head and batch dims

def _create_custom_4d_mask(self, sequence_length, dtype, device, batch_size):
    """Compute the full mask by expanding the cached base to batch size."""
    base = self.compute_mask_base(sequence_length, dtype, device)
    return base.expand(batch_size, -1, -1, -1)
```

评论区精华

Review 中，Isotr0py 对缓存有效性提出质疑，担心 sequence_length 变化导致缓存低效；LiuLi1998 回应实际只有 288 和 512 两个值，缓存有界。随后 Isotr0py 建议改用 @classmethod+@lru_cache，并提交 commit 实现。最终方案保留了 dtype 和 device 作为缓存 key，以兼容不同配置。

- 缓存 key 范围和实现方式 (design): 保留 (sequence_length, dtype, device) 作为缓存 key ，采用 @lru_cache，LiuLi1998 测试通过。
- 批次不变性假设验证 (correctness): 通过基准测试和 profile 确认正确性。

风险与影响

- 风险：风险较低。缓存 key 包括 (sequence_length, dtype, device)，典型部署下组合有限；lru_cache 的 maxsize=8 限制了条目数。无动态控制流变化，正确性通过基准测试验证。未修改 batch 维度的计算，扩展不会引入新回归。若未来模型变种导致更多 sequence_length 值，需注意缓存命中率，但当前场景无虞。
- 影响：直接影响使用 DeepSeek-OCR-2 模型的用户，TTFT 显著降低（p50 降 46%），吞吐量提升 83%。对其他模型无影响。团队维护成本极低——仅一个文件改动，无新增依赖，部署无需额外配置。
- 风险标记：缓存假设依赖固定 token_type_ids 模式，仅针对 OCR-2 模型优化

关联脉络

- 暂无明显关联 PR