PR #25773 完整报告

sgl-project/sglang

Add fused_rope and for xpu

合并时间：2026-06-03 09:41

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25773

执行摘要

- 一句话：XPU 融合 RoPE 内核提升解码性能
- 推荐动作：值得精读，了解 XPU 上基于 head_size 的 kernel 选择策略和条件分支设计。

功能与动机

fused_qk_rope 可以减少内核启动次数，利用编译时常量优化向量化加载 / 存储，从而提升 XPU 平台 Rotary Embedding 计算性能。详见 review 中作者解释。

实现拆解

1. 在条件导入块中添加对 fused_qk_rope_with_cos_sin_cache_inplace 的导入：当平台为 xpu 时，从 sgl_kernel 导入该函数。
2. 重写 forward_xpu 方法：根据 head_size 判断是否在 [128, 256, 512] 中，若是则走融合路径，否则回退到原有的 torch.ops.sgl_kernel.rotary_embedding 调用。
3. 融合路径中对 query 和 key 进行 reshape 和 rotary_dim 切片，然后原地调用融合内核；回退路径保持原有逻辑。 未涉及测试、配置或部署配套更改。

关键文件：

- python/sglang/srt/layers/rotary_embedding/base.py（模块 旋转嵌入；类别 source；类型 core-logic；符号 forward_xpu, fused_qk_rope_with_cos_sin_cache_inplace）：唯一修改文件，添加 fused_qk_rope 内核使用，重写 forward_xpu 方法

关键符号：forward_xpu, fused_qk_rope_with_cos_sin_cache_inplace

关键源码片段

python/sglang/srt/layers/rotary_embedding/base.py

唯一修改文件，添加 fused_qk_rope 内核使用，重写 forward_xpu 方法

```
def forward_xpu(
    self,
    positions: torch.Tensor,
    query: torch.Tensor,
    key: torch.Tensor,
    offsets: Optional[torch.Tensor] = None,
```

```
    fused_set_kv_buffer_arg: Optional[FusedSetKVBufferArg] = None,
) -> Tuple[torch.Tensor, torch.Tensor]:
    # fused_set_kv_buffer_arg 在 xpu 实现中不支持
    assert (
        fused_set_kv_buffer_arg is None
    ), "fused_set_kv_buffer_arg is not supported for xpu implementation"

    # 处理 offsets（用于多批次位置偏移）
    positions = torch.add(positions, offsets) if offsets is not None else positions

    # 确保 cos/sin cache 的 dtype 与 query 一致
    self._match_cos_sin_cache_dtype(query)

    # Fused_qk_rope 只支持对齐的 head_size（128, 256, 512）
    if self.head_size in [128, 256, 512]:
        num_tokens = positions.size(0)
        # 将 query 和 key 重塑为 [num_tokens, -1, head_size] 以分离 head 维度
        q_rope = query.view(num_tokens, -1, self.head_size)
        k_rope = key.view(num_tokens, -1, self.head_size)
        # 如果 rotary_dim 小于 head_size，只取前 rotary_dim 部分
        if self.head_size != self.rotary_dim:
            q_rope = q_rope[..., : self.rotary_dim]
            k_rope = k_rope[..., : self.rotary_dim]
        # 原地调用融合 kernel，避免额外内存分配
        fused_qk_rope_with_cos_sin_cache_inplace(
            q_rope,
            k_rope,
            self.cos_sin_cache,
            positions,
            self.rotary_dim,
            self.is_neox_style,
        )
        return query, key
    else:
        # 对于不支持的 head_size，回退到通用 rotary_embedding kernel
        return torch.ops.sgl_kernel.rotary_embedding(
            positions,
            query,
            key,
            self.head_size,
            self.cos_sin_cache,
            self.is_neox_style,
        )
```

评论区精华

> mingfeima: the logic here is not clear. so where does the performance benefit comes from? inplace? gaopengff: This fused_qk_rope is a jit kernel in cuda, which means it could use constant value of is_neox, rope_dim to launch kernel. The

> load/store vectorized size is tuned from rope_dim. Also, it launched fewer kernels compared to rotary_embedding. For xpu version, I have a tuned vector size load/store PR: https://github.com/sgl-project/sgl-kernel-xpu/pull/221.
>
> gemini-code-assist[bot]: Creating q_weight and k_weight tensors of ones on every forward pass introduces unnecessary overhead. These should ideally be pre-allocated as buffers. gaopengff: Use new method without creating new tensors.
>
> gemini-code-assist[bot]: The forward_xpu method is defined twice in MRotaryEmbedding class. 本 PR 未涉及 mrope.py 的修改，该重复定义问题未在本 PR 中处理。

- 性能收益来源分析 (performance): 解释合理，性能收益主要来自内核融合和向量化优化。
- 重复定义 forward_xpu (design): 本 PR 未涉及 mrope.py 的修改，该问题未在本 PR 中处理。
- 临时张量分配和 cat 开销 (performance): 作者已修改代码，移除了临时张量和 cat 操作，采用 view 和索引切片，性能问题已解决。

风险与影响

- 风险：只有 base.py 一个文件变更，且仅影响 XPU 路径的 forward_xpu；新增的分支逻辑与原有回退路径功能等效，但缺少显式测试覆盖，若融合内核在特定 head_size 下有正确性问题则可能导致静默错误。回退路径保留，可降低部分风险。
- 影响：仅影响 XPU 平台（Intel GPU 等）的推理延迟，对解码性能有正面提升；对其他平台无影响。用户无需修改代码即可受益。
- 风险标记：缺少测试覆盖，仅影响 XPU 路径

关联脉络

- 暂无明显关联 PR