

PR #37888 完整报告

vllm-project/vllm

[XPU] Enable multiple key kernels for sparse attention

合并时间: 2026-05-22 12:02

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/37888>

执行摘要

- 一句话: XPU 稀疏 attention 内核迁移至统一 custom op
- 推荐动作: 值得精读, 特别是以下设计决策:
 - 如何通过 `_custom_ops` 统一平台差异, 使上层代码不再关心后端。
 - 移除 Python fallback 的风险与收益权衡。
 - 对 vLLM IR 的讨论揭示了未来的架构演进方向。

功能与动机

PR body 明确指出: “Reduce duplicated backend-specific Python paths and rely on consolidated custom-op dispatch, while ensuring XPU sparse indexer uses the intended kernels from `vllm-xpu-kernels`。”此外, review 中 wuxun-zhang 建议 “enable `topk_per_row` and `indexer_quant_cache` kernels for DeepSeek V3.2”, 说明本 PR 是 XPU 支持 DeepSeek V3.2 的关键一步。

实现拆解

1. 在 `vllm/_custom_ops.py` 中添加共享包装: 新增 `top_k_per_row_prefill` 和 `top_k_per_row_decode` 函数, 直接调用 `torch.ops._C.top_k_per_row_prefill/decode`, 提供与 CUDA 一致的接口供上层统一调用。
2. 在 `vllm/_xpu_ops.py` 中添加 XPU 专用包装: 新增 `_xpu_fp8_mqa_logits_impl/fake` 和 `_xpu_fp8_paged_mqa_logits_impl/fake`, 分别调用 `torch.ops._xpu_C.fp8_mqa_logits` 和 `torch.ops._xpu_C.fp8_paged_mqa_logits`。fake 函数仅在非 XPU 平台或 tracing 时使用, 返回空 tensor。
3. 移除 `xpu_ops` 类中的 Python fallback: 删除了 `indexer_k_quant_and_cache` (内含纯 Python 实现的 `group_quant_torch`)、`cp_gather_indexer_k_quant_cache`、`top_k_per_row_prefill` 和 `top_k_per_row_decode`, 这些逻辑现已由 `vllm-xpu-kernels` 中的 C++ 内核替代。
4. 统一 `sparse_attn_indexer.py` 的导入与路径: 将条件导入 (`if cuda_alike: from ... import ops elif is_xpu: from ... import xpu_ops`) 改为统一的 `from vllm import _custom_ops as ops`。在 `prefill` 阶段, XPU 路径直接调用 `torch.ops.vllm.xpu_fp8_mqa_logits` (而非 DeepGEMM 的 `fp8_fp4_mqa_logits`), 并加入 `RuntimeError` 防护不支持 FP4 Q。在 `decode` 阶段, XPU 路径调用

`torch.ops.vllm.xpu_fp8_paged_mqa_logits`, 并对 `seq_lens` 做维度兼容。

5. 新增 `forward_xpu` 方法: 在 `SparseAttnIndexer` 类中单独添加 `forward_xpu`, 但后续 review 讨论中决定改为复用 `forward_cuda` 以减少代码重复。

关键文件:

- `vllm/_xpu_ops.py` (模块 核心操作; 类别 source; 类型 core-logic; 符号 `_xpu_fp8_mqa_logits_impl`, `_xpu_fp8_mqa_logits_fake`, `_xpu_fp8_paged_mqa_logits_impl`, `_xpu_fp8_paged_mqa_logits_fake`): 核心变更文件: 新增 FP8 MQA logits 包装, 移除大量 Python fallback, 是 XPU 内核迁移的主阵地。
- `vllm/model_executor/layers/sparse_attn_indexer.py` (模块 稀疏注意力; 类别 source; 类型 core-logic; 符号 `forward_xpu`): 上层调用逻辑统一, 移除平台条件分支, 引入 XPU 内核调用路径。
- `vllm/_custom_ops.py` (模块 自定义操作; 类别 source; 类型 core-logic; 符号 `top_k_per_row_prefill`, `top_k_per_row_decode`): 新增 `top_k_per_row_prefill` 和 `top_k_per_row_decode` 共享包装, 使上层可统一调用。

关键符号: `_xpu_fp8_mqa_logits_impl`, `_xpu_fp8_mqa_logits_fake`, `_xpu_fp8_paged_mqa_logits_impl`, `_xpu_fp8_paged_mqa_logits_fake`, `indexer_k_quant_and_cache`, `group_quant_torch`, `cp_gather_indexer_k_quant_cache`, `top_k_per_row_prefill`, `top_k_per_row_decode`, `forward_xpu`

关键源码片段

`vllm/_xpu_ops.py`

核心变更文件: 新增 FP8 MQA logits 包装, 移除大量 Python fallback, 是 XPU 内核迁移的主阵地。

```
# vllm/_xpu_ops.py (head 版本)
```

```
# --- 新增: FP8 MQA logits 包装 ---
```

```
def _xpu_fp8_mqa_logits_impl(
    q: torch.Tensor,
    k_quant: torch.Tensor,
    k_scale: torch.Tensor,
    weights: torch.Tensor,
    cu_seqlen_ks: torch.Tensor,
    cu_seqlen_ke: torch.Tensor,
) -> torch.Tensor:
    # 直接调用 xpu 后端的 C++ 内核
    return torch.ops._xpu_C.fp8_mqa_logits(
        q, k_quant, k_scale, weights, cu_seqlen_ks, cu_seqlen_ke
    )
```

```
def _xpu_fp8_mqa_logits_fake(
    q: torch.Tensor,
    k_quant: torch.Tensor,
    k_scale: torch.Tensor,
```

```

weights: torch.Tensor,
cu_seqlen_ks: torch.Tensor,
cu_seqlen_ke: torch.Tensor,
) -> torch.Tensor:
    # fake 实现用于 tracing / 非 XPU 环境
    return torch.empty(
        (q.shape[0], k_quant.shape[0]), dtype=torch.float32, device=q.device
    )

# --- 新增: FP8 Paged MQA logits 包装 ---
def _xpu_fp8_paged_mqa_logits_impl(
    q: torch.Tensor,
    kv_cache: torch.Tensor,
    weights: torch.Tensor,
    context_lens: torch.Tensor,
    block_tables: torch.Tensor,
    schedule_metadata: torch.Tensor,
    max_model_len: int,
) -> torch.Tensor:
    return torch.ops._xpu_C.fp8_paged_mqa_logits(
        q, kv_cache, weights, context_lens, block_tables, schedule_metadata, max_model_len
    )

# --- 已删除的大段 Python fallback 包括:
# indexer_k_quant_and_cache (内含 group_quant_torch 纯 Python 量化和缓存逻辑)
# cp_gather_indexer_k_quant_cache (纯 Python block-gather 实现)
# top_k_per_row_prefill / top_k_per_row_decode (纯 Python topk 逻辑)

```

vllm/model_executor/layers/sparse_attn_indexer.py

上层调用逻辑统一，移除平台条件分支，引入 XPU 内核调用路径。

```

# vllm/model_executor/layers/sparse_attn_indexer.py (head 版本部分摘录)

# 导入统一: 直接使用 _custom_ops 作为 ops
from vllm import _custom_ops as ops
# 不再有条件导入 (已删除 is_cuda_alike/is_xpu 判断)

# prefill 循环中的 XPU 路径示例:
for chunk in prefill_metadata.chunks:
    # ... 省略 k_quant/k_scale 准备 ...
    if current_platform.is_xpu():
        # XPU 路径: 不支持 FP4 Q, 直接报错
        if q_scale_slice is not None:
            raise RuntimeError("XPU fp8_mqa_logits does not support FP4 Q")
        # 直接调用注册的自定义 op (通过 torch.ops.vllm 注册)
        logits = torch.ops.vllm.xpu_fp8_mqa_logits(
            q_slice_cast, k_quant_cast, k_scale_cast,
            weights[chunk.token_start : chunk.token_end],
            chunk.cu_seqlen_ks, chunk.cu_seqlen_ke,

```

```

    )
else:
    # 非 XPU 路径 (CUDA) 继续使用 DeepGEMM
    logits = fp8_fp4_mqa_logits(...)

    # top-k 操作已经统一到 ops.top_k_per_row_prefill
    ops.top_k_per_row_prefill(
        logits, chunk.cu_seqlen_ks, chunk.cu_seqlen_ke,
        topk_indices, num_rows, logits.stride(0), logits.stride(1), topk_tokens
    )

```

vllm/_custom_ops.py

新增 top_k_per_row_prefill 和 top_k_per_row_decode 共享包装，使上层可统一调用。

vllm/_custom_ops.py (head 版本新增片段)

```

def top_k_per_row_prefill(
    logits: torch.Tensor,
    cu_seqlen_ks: torch.Tensor,
    cu_seqlen_ke: torch.Tensor,
    raw_topk_indices: torch.Tensor,
    num_rows: int,
    stride0: int,
    stride1: int,
    topk_tokens: int,
) -> None:
    # 统一包装 CUDA top_k_per_row_prefill 内核
    torch.ops._C.top_k_per_row_prefill(
        logits, cu_seqlen_ks, cu_seqlen_ke, raw_topk_indices,
        num_rows, stride0, stride1, topk_tokens
    )

```

```

def top_k_per_row_decode(
    logits: torch.Tensor,
    next_n: int,
    seq_lens: torch.Tensor,
    raw_topk_indices: torch.Tensor,
    num_rows: int,
    stride0: int,
    stride1: int,
    topk_tokens: int,
) -> None:
    torch.ops._C.top_k_per_row_decode(
        logits, next_n, seq_lens, raw_topk_indices,
        num_rows, stride0, stride1, topk_tokens
    )

```

评论区精华

- gemini-code-assist[bot] 指出两个潜在问题：一是 `indexer_k_quant_and_cache` 中使用了错误的命名空间 `_C_cache_ops`，应为 `_C`；二是 `top_k_per_row_prefill` 参数名 `strdide1` 拼写错误。这些已在后续提交中修复。
- jikunshang 建议将 `top_k` 操作移到 `_custom_ops.py`，使 XPU 和 CUDA 共用同一路径，避免条件分支。最终采纳。
- wuxun-zhang 指出 `cp_gather_indexer_k_quant_cache` 已在 `vllm-xpu-kernels 0.1.5` 中支持，应移除 Python fallback。作者照做。
- yma11 质疑 `forward_xpu` 与 `forward_cuda` 的代码重复，建议用 vLLM IR 统一。作者短期内接受用 `activation.py` 中的模式简化，长期期待 IR 解决。
- xinyu-intel 与 ProExpertProg 讨论 vLLM IR 如何消除平台特定分支，最终结论是 XPU 可注册自己的 IR kernel，将来只保留一个统一的 `forward`。
- 自定义 op 命名空间 `_C_cache_ops` 疑似笔误 (correctness): 作者修复了该问题，后续提交中改正。
- `top_k_per_row_prefill` 参数名 `strdide1` 拼写错误 (style): 作者修复了拼写，统一为 `stride1`。
- `top_k` 操作是否应移至 `_custom_ops.py` (design): 作者将 `top_k_per_row_prefill` 和 `top_k_per_row_decode` 移入 `_custom_ops.py`，上层统一使用 `ops` 调用。
- `cp_gather_indexer_k_quant_cache` 是否仍有必要保留 Python fallback (design): 作者删除了 `xpu_ops.cp_gather_indexer_k_quant_cache`，统一使用 C++ 内核。
- `forward_xpu` 方法代码重复问题 (design): 作者改为复用 `forward_cuda`（参见 `activation.py` 模式），并期待 vLLM IR 最终统一。
- vLLM IR 是否能消除平台相关分支 (design): 当前仍用条件分支，未来 IR 成熟后可以简化。

风险与影响

- 风险：
 - 依赖外部包版本：本 PR 需要 `vllm-xpu-kernels >= 0.1.8` 才能提供 `fp8_mqa_logits` 和 `fp8_paged_mqa_logits`；若版本不满足，运行时将出现 `AttributeError`。
 - 移除大量 Python fallback 可能覆盖不足：原先 `indexer_k_quant_and_cache` 中有完整的量化逻辑 (`group_quant_torch`)，删除后若新内核有隐藏 bug，无降级路径。
 - 新加的 `RuntimeError` 检查：在 XPU 路径中若 `q_scale_slice` is not None 直接抛出 `RuntimeError`，禁止使用 FP4 Q；这可能打断某些尚未暴露的使用场景。
 - 无新增测试文件：仅依赖 CI 中的 MLA 相关测试，边界情况可能未被覆盖。
 - 潜在性能风险：`seq_lens` 维度兼容处理 (`seq_lens[:, -1].contiguous()`) 若 `shape` 不符合预期可能出错。
- 影响：
 - 用户影响：XPU (Intel GPU) 用户现在可以运行 DeepSeek V3.2 模型的稀疏 attention 路径，且性能依赖 `vllm-xpu-kernels` 的优化。CUDA 用户无感知。

- 系统影响：减少了 `_xpu_ops.py` 中 245 行冗余代码，提升了可维护性；统一的 `_custom_ops` 接口使后续添加新内核更简单。
- 团队影响：XPU 后端开发流程与主仓库对齐，新功能只需在 `_xpu_ops.py` 中添加包装即可，不需修改上层调用逻辑。
- 风险标记：依赖外部包版本，Python fallback 完全移除，XPU 不支持 FP4 Q 时硬中断，无新增测试文件，`seq_lens` 维度兼容改动可能出错

关联脉络

- PR #37968 [XPU] Add fp8_mqa_logits and fp8_paged_mqa_logits kernels: 本 PR 等待该 PR 的 XPU 内核合并后才能使用 `fp8_mqa_logits` 和 `fp8_paged_mqa_logits`。
- PR #39968 Dependency PR for vllm-xpu-kernels version bump: PR body 中明确标记为依赖，需要 `vllm-xpu-kernels` 0.1.8 发布。