

PR #37622 完整报告

sgl-project/sglang

perf(lfm2): fuse gating and short convolution on SM90

合并时间: 2026-09-05 21:52

原文链接: <http://prhub.com.cn/sgl-project/sglang/pull/37622>

执行摘要

- 一句话: 融合 LFM2 门控与短卷积内核, 消除预填充冗余操作, 提升吞吐约 3%。
- 推荐动作: 这是一个高质量、目标明确的性能优化 PR。其价值在于: 1) 针对具体性能瓶颈 (多次内核启动与转置) 给出了高效的解决方案; 2) 实现了严格的条件调度, 确保优化安全且不影响通用路径; 3) 提供了全面、独立的测试用例。建议对高性能计算、Triton 内核或 SGLang 模型性能优化感兴趣的相关工程师精读核心内核实现 (lfm_short_conv.py), 学习其融合计算和状态管理的设计思路。

功能与动机

LFM2-MoE 的短卷积路径当前需要依次计算 $B * x$, 将结果转置以适应通用 causal-convolution 内核, 然后再计算 $C * \text{conv}(B * x)$ 。在 BF16 TP1 的 LFM2.5-8B-A1B 服务配置下, 这些独立的内核启动和转置操作构成了可测量的预填充瓶颈。PR body 明确指出需要“fuse both gates with the width-3 depthwise convolution”以提升性能。

实现拆解

1. 核心内核实现: 在 `python/sglang/kernels/ops/mamba/lfm_short_conv.py` 中新增两个 `@triton.jit` 函数 `_lfm_short_conv_prefill_kernel` 和 `_lfm_short_conv_decode_kernel`。它们直接在 SM90 GPU 上融合了 B 门控乘法 ($B * x$)、3 点深度卷积和 C 门控乘法 ($C * \text{conv}$), 同时保留了两个 BF16 数据物化点并支持原地递归状态更新。
2. 调度逻辑与接口: 同一文件中添加了 `can_use_fused_lfm_short_conv` 和 `can_dispatch_fused_lfm_short_conv` 函数, 用于严格检查输入是否满足融合条件 (如 SM90 架构、BF16 精确类型、无偏置、宽度为 3、隐藏维度为 2048)。
`fused_lfm_short_conv_prefill` 和 `fused_lfm_short_conv_decode` 是公开的 Python API。
3. 模型集成: 在 `python/sglang/srt/models/lfm2_moe.py` 的 `Lfm2MoeShortConvLayer.forward` 方法中, 引入了对 `can_dispatch_fused_lfm_short_conv` 的调用。当条件满足时 (decode 或 prefill), 会优先调用融合内核; 否则回退到原有的 `causal_conv1d_fn/causal_conv1d_update` 实现。
4. 测试覆盖: 新增 `test/registered/kernels/ops/mamba/test_lfm_short_conv.py`, 包含 4 个测试用例, 全面覆盖了变量序列预填充、解码 (含填充槽位)、CUDA Graph 重放以及调度守卫逻辑的正确性。

关键文件:

- python/sglang/kernels/ops/mamba/lfm_short_conv.py (模块 Triton 内核; 类别 infra; 类型 infrastructure; 符号 _lfm_short_conv_prefill_kernel, _lfm_short_conv_decode_kernel, can_use_fused_lfm_short_conv, can_dispatch_fused_lfm_short_conv) : 包含新增的 SM90 专用融合 Triton 内核及调度逻辑, 是本次性能优化的核心实现。
- python/sglang/srt/models/lfm2_moe.py (模块 模型层; 类别 source; 类型 data-contract) : LFM2-MoE 模型的主文件, 其短卷积层集成融合内核的调度逻辑, 是优化生效的入口。
- test/registered/kernels/ops/mamba/test_lfm_short_conv.py (模块 内核测试; 类别 test ; 类型 test-coverage; 符号 _inputs, make, _reference_prefill, _reference_decode) : 新增的综合测试文件, 验证融合内核在各种场景 (变量序列、填充槽位、CUDA Graph、调度条件) 下的正确性, 是质量保障的关键。

关键符号: _lfm_short_conv_prefill_kernel, _lfm_short_conv_decode_kernel, can_use_fused_lfm_short_conv, can_dispatch_fused_lfm_short_conv, fused_lfm_short_conv_prefill, fused_lfm_short_conv_decode

关键源码片段

python/sglang/kernels/ops/mamba/lfm_short_conv.py

包含新增的 SM90 专用融合 Triton 内核及调度逻辑, 是本次性能优化的核心实现。

```
# python/sglang/kernels/ops/mamba/lfm_short_conv.py
# 融合内核的公开调度入口, 包含严格的条件检查。
def can_dispatch_fused_lfm_short_conv(
    b: torch.Tensor,
    c: torch.Tensor,
    x: torch.Tensor,
    weight: torch.Tensor,
    bias: torch.Tensor | None,
    state: torch.Tensor,
) -> bool:
    """Return whether this call is in the end-to-end-qualified serving domain."""
    return (
        b.ndim == 2
        and b.shape[1] == 2048 # 仅针对隐藏维度为 2048 的模型
        and can_use_fused_lfm_short_conv(b, c, x, weight, bias, state)
    )

def can_use_fused_lfm_short_conv(
    b: torch.Tensor,
    c: torch.Tensor,
    x: torch.Tensor,
    weight: torch.Tensor,
    bias: torch.Tensor | None,
    state: torch.Tensor,
) -> bool:
    """Return whether the exact LFM2 BF16 width-3 contract is satisfied."""
```

```

# 检查: SM90 架构、非 ROCm、所有张量在相同 CUDA 设备、
# 输入为 BF16 的 [tokens, dim] 格式且列连续、权重为 BF16 的 [dim, 3] 且列连续、
# 无偏置、状态为 BF16 的 [slots, dim, 2]。
return (
    not _DISABLE_LFM_FUSED_CONV
    and b.is_cuda
    and torch.version.hip is None
    and torch.cuda.get_device_capability(b.device) == (9, 0)
    and c.is_cuda
    and x.is_cuda
    and weight.is_cuda
    and state.is_cuda
    and b.device == c.device == x.device == weight.device == state.device
    and b.dtype == c.dtype == x.dtype == torch.bfloat16
    and b.shape == c.shape == x.shape
    and b.ndim == 2
    and b.stride(1) == c.stride(1) == x.stride(1) == 1
    and weight.dtype == b.dtype
    and weight.ndim == 2
    and weight.shape == (b.shape[1], 3)
    and weight.stride(1) == 1
    and bias is None
    and state.dtype == b.dtype
    and state.ndim == 3
    and state.shape[1:] == (b.shape[1], 2)
)

```

python/sglang/srt/models/lfm2_moe.py

LFM2-MoE 模型的主文件，其短卷积层集成融合内核的调度逻辑，是优化生效的入口。

```

# python/sglang/srt/models/lfm2_moe.py
# 在 Lfm2MoeShortConvLayer.forward 中集成融合内核。
def forward(
    self,
    hidden_states: torch.Tensor,
    forward_batch: ForwardBatch,
) -> torch.Tensor:
    if forward_batch.forward_mode.is_idle():
        return hidden_states

    meta = get_attn_backend().conv_state_metadata(self.layer_idx, forward_batch)
    conv_state = meta.layer_cache.conv[0]

    proj, _ = self.in_proj(hidden_states)
    B_gate, C_gate, x = proj.chunk(3, dim=-1)
    # 检查是否满足融合内核的调度条件
    use_fused = can_dispatch_fused_lfm_short_conv(
        B_gate, C_gate, x, self.conv_weight, self.conv_bias, conv_state
    )

```

```

if forward_batch.forward_mode.is_decode() and use_fused:
    # 条件满足时, 使用融合解码内核
    gated_conv_out = fused_lfm_short_conv_decode(
        B_gate, C_gate, x, self.conv_weight, conv_state, meta.cache_indices
    )
elif forward_batch.forward_mode.is_decode():
    # 回退到原始解码路径
    Bx = B_gate * x
    conv_out = causal_conv1d_update(Bx, conv_state, self.conv_weight,
                                     self.conv_bias, ...)
    gated_conv_out = C_gate * conv_out
# ... 其他分支 (target_verify, prefill) 类似
elif (
    use_fused
    and meta.query_start_loc is not None
    and meta.has_initial_state is not None
    and forward_batch.extend_seq_lens_cpu
):
    # 条件满足时, 使用融合预填充内核
    gated_conv_out = fused_lfm_short_conv_prefill(
        B_gate, C_gate, x, self.conv_weight, conv_state,
        meta.query_start_loc, meta.cache_indices, meta.has_initial_state,
        max(forward_batch.extend_seq_lens_cpu)
    )
else:
    # 回退到原始预填充路径
    Bx = B_gate * x
    Bx_t = Bx.transpose(0, 1).contiguous()
    conv_out = causal_conv1d_fn(Bx_t, self.conv_weight, ...)
    gated_conv_out = C_gate * conv_out

output, _ = self.out_proj(gated_conv_out)
return output

```

评论区精华

PR 评论区未涉及技术设计或实现的深入讨论。作者 BBuf 主要进行了两次 CI 状态的确认与标注：一次是使用 `/tag-and-rerun-ci` 触发测试，另一次是确认新增的测试文件 `test_lfm_short_conv.py` 在 CI 上成功通过。

- 暂无高价值评论线程

风险与影响

- 风险：

1. 数值精度风险：融合内核在中间计算中使用 FP32 累加后转换回 BF16，可能导致与原始路径存在微小的数值差异。测试中通过设置 `rtol=0.02`, `atol=0.05` 的容差验证活跃令牌输出，并断言递归状态位完全一致来缓解此风险。

2. 平台与配置限制：融合逻辑仅在 SM90 (Hopper) 架构、BF16 精确类型、隐藏维度恰好为 2048、无偏置等非常具体的条件下激活。这种窄调度策略降低了引入全局性错误的风险，但意味着在其他 GPU 或配置下性能无提升，且维护两条代码路径（融合与原始）增加了复杂性。

- 影响：

1. 用户影响：直接提升使用 LFM2-MoE 模型（特别是 LFM2.5-8B-A1B）在 Hopper GPU 上的服务吞吐量。根据报告，预填充卷积段加速 2.95 倍，端到端服务吞吐（1024 输入 / 256 输出，32 并发）提升 3.15%。
2. 系统影响：对系统无破坏性影响。变更被严格的条件守卫，不影响不符合条件的模型或硬件路径。新增内核是纯计算模块，不改变模型权重、API 或对外数据契约。
3. 团队影响：为在 SGLang 中实现类似的高性能内核融合提供了范例，展示了如何通过精确的条件检查将优化限定在安全范围。 - 风险标记：数值精度微差，SM90 平台限制，双代码路径维护

关联脉络

- PR #38039 fix(mamba): unify causal_conv1d col* dtype to x (MiniCPM-V-4.6 GDN prefill bf16/fp16 mismatch): 此 PR 修复了 causal_conv1d Triton 内核中的类型不匹配崩溃问题。本 PR 中的融合内核实现了新的计算路径，但同样依赖于底层的 causal_conv1d 作为后备，因此 38039 的修复为本 PR 提供了更稳定的基础。
- PR #38085 fix(moe): cast filtered-activation expert_ids to int32 for torch.compile: 此 PR 修复了 MoE 路径下的一个类型问题。虽然与本 PR 的卷积融合不直接相关，但都属于模型核心计算路径的优化与修复，反映了对模型稳定性和性能的持续改进。