PR #24048 完整报告

sgl-project/sglang

[VLM] Optimize Gemma4 VLM with PCG and fuse RMSNorm + residual add + scalar

合并时间：2026-05-05 00:36

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24048

执行摘要

- 一句话：融合 kernel 与 PCG 提升 Gemma4 预填充性能
- 推荐动作：建议精读 gemma4_fused_ops.py 中的 kernel 融合策略（减少 launch overhead 的典型模式）和 gemma4_mm.py 中 model 属性与 __setattr__ 的设计（在不破坏 state_dict 前提下兼容 PCG 框架），这些模式对类似优化有借鉴价值。

功能与动机

Optimize Gemma4 26B-A4B prefill performance through two complementary approaches:

1. Fused Triton kernels for Gemma4 decoder layers — Reduces kernel launch overhead by fusing multiple operations into single kernels. 2. Enable Piecewise CUDA Graph (PCG) for VLM models — Fixes PCG support for multimodal models that use self.language_model instead of self.model to reference their text backbone.

实现拆解

1. 新增融合 kernel：在 gemma4_fused_ops.py 中定义 _gemma_dual_rmsnorm_residual_kernel 及 wrapper gemma_dual_rmsnorm_residual_scalar，将 MoE 分支后两个密度归一化（post_feedforward_layernorm_1、post_feedforward_layernorm_2）与最终归一化（post_feedforward_layernorm）、残差加和标量乘融合为单一 Triton kernel。
2. 集成到 Gemma4 解码器：修改 gemma4_causal.py 的 Gemma4DecoderLayer.forward()，在 MoE 块中直接调用融合 kernel，替代原有的逐层 norm 调用和临时张量分配。
3. PCG 兼容性适配：在 gemma4_mm.py 的 Gemma4ForConditionalGeneration 中添加 model 属性（别名为 language_model）和 __setattr__ 阻断对 model 的赋值，以通过 PCG 门控检查 hasattr(model, "model") 同时防止 state_dict 键重复。
4. 模型运行器扩展：在 model_runner.py 中扩展 resolve_language_model() 支持 language_model 属性；在 init_piecewise_cuda_graphs() 中增加灵活的 layers 解析逻辑，适配 language_model 直接包含 layers 的架构。
5. 图运行器适配：在 piecewise_cuda_graph_runner.py 中更新 patch_model 的目标解析，支持 language_model 作为模型主干的情况。

关键文件：

- python/sglang/srt/layers/gemma4_fused_ops.py（模块 融合内核；类别 source；类型 core-logic；符号 _gemma_dual_rmsnorm_residual_kernel, gemma_dual_rmsnorm_residual_scalar）：新增核心融合 kernel _gemma_dual_rmsnorm_residual_kernel 和 wrapper，是性能提升的关键。
- python/sglang/srt/models/gemma4_mm.py（模块 多模态模型；类别 source；类型 data-contract；符号 model, setattr）：添加 model 属性和 __setattr__以绕过 PCG 门控并防止 state_dict 污染，是 PCG 启用的关键。
- python/sglang/srt/models/gemma4_causal.py（模块 解码器层；类别 source；类型 data-contract）：修改 DecoderLayer.forward() 使用融合 kernel，是性能提升的落地。
- python/sglang/srt/model_executor/model_runner.py（模块 模型运行器；类别 source；类型 data-contract）：扩展 resolve_language_model 和 PCG 初始化逻辑，支持 VLM 架构。
- python/sglang/srt/model_executor/piecewise_cuda_graph_runner.py（模块 图编译；类别 source；类型 data-contract）：适配 language_model 直接包含 layers 的情况。

关键符号：_gemma_dual_rmsnorm_residual_kernel, gemma_dual_rmsnorm_residual_scalar, Gemma4ForConditionalGeneration.model, Gemma4ForConditionalGeneration.setattr, resolve_language_model, init_piecewise_cuda_graphs

关键源码片段

python/sglang/srt/layers/gemma4_fused_ops.py

新增核心融合 kernel _gemma_dual_rmsnorm_residual_kernel 和 wrapper，是性能提升的关键。

```
# gemma4_fused_ops.py
@triton.jit
def _gemma_dual_rmsnorm_residual_kernel(
    X1_ptr, W1_ptr, X2_ptr, W2_ptr, W3_ptr,
    Residual_ptr, Scalar_ptr, Out_ptr,
    stride_x1, stride_x2, stride_r, stride_o,
    N, eps1, eps2, eps3,
    BLOCK_SIZE: tl.constexpr,
):
    # 将 MoE 块尾部的 3 个 RMSNorm + 残差加 + 标量乘融合为单次 kernel 调用
    row = tl.program_id(0)
    cols = tl.arange(0, BLOCK_SIZE)
    mask = cols < N

    x1 = tl.load(X1_ptr + row * stride_x1 + cols, mask=mask, other=0.0).to(tl.float32)
    w1 = tl.load(W1_ptr + cols, mask=mask, other=0.0).to(tl.float32)
    x2 = tl.load(X2_ptr + row * stride_x2 + cols, mask=mask, other=0.0).to(tl.float32)
    w2 = tl.load(W2_ptr + cols, mask=mask, other=0.0).to(tl.float32)
    w3 = tl.load(W3_ptr + cols, mask=mask, other=0.0).to(tl.float32)
    r = tl.load(Residual_ptr + row * stride_r + cols, mask=mask, other=0.0).to(tl.float32)
```

```
    var1 = tl.sum(x1 * x1, axis=0) / N
    norm1 = x1 * tl.rsqrt(var1 + eps1) * w1 # 第一个 RMSNorm

    var2 = tl.sum(x2 * x2, axis=0) / N
    norm2 = x2 * tl.rsqrt(var2 + eps2) * w2 # 第二个 RMSNorm

    combined = norm1 + norm2

    var3 = tl.sum(combined * combined, axis=0) / N
    norm3 = combined * tl.rsqrt(var3 + eps3) * w3 # 第三个 RMSNorm（融合后）

    scalar = tl.load(Scalar_ptr).to(tl.float32)
    out = (norm3 + r) * scalar # 残差加 + 标量乘

    tl.store(Out_ptr + row * stride_o + cols, out.to(x1.dtype), mask=mask)


def gemma_dual_rmsnorm_residual_scalar(
    x1, weight1, x2, weight2, weight3, residual, scalar,
    eps1=1e-6, eps2=1e-6, eps3=1e-6,
):
    # 前置检查：确保 x1 是二维且最后维连续，避免 Triton kernel 越界
    assert x1.dim() == 2 and x1.stride(-1) == 1
    M, N = x1.shape
    BLOCK_SIZE = triton.next_power_of_2(N)
    out = torch.empty_like(x1)

    _gemma_dual_rmsnorm_residual_kernel[(M,)](
        x1, weight1, x2, weight2, weight3, residual, scalar, out,
        x1.stride(0), x2.stride(0), residual.stride(0), out.stride(0),
        N, eps1, eps2, eps3,
        BLOCK_SIZE=BLOCK_SIZE,
    )
    return out
```

python/sglang/srt/models/gemma4_mm.py

添加 model 属性和 __setattr__以绕过 PCG 门控并防止 state_dict 污染，是 PCG 启用的关键。

```
# gemma4_mm.py
class Gemma4ForConditionalGeneration(nn.Module):
    # ...
    @property
    def model(self):
        # 将 .model 别名为 .language_model，使得外层检查 hasattr(model, "model")
        # 通过，同时避免注册重复子模块导致 state_dict 键重复
        return self.language_model

    def __setattr__(self, name, value):
        # 阻断对 "model" 的直接赋值，防止 runner 的
```

```
        # self.model.model = resolve_language_model(self.model)
        # 意外注册 nn.Module 到 _modules，从而污染 state_dict
        if name == "model":
            return
        super().__setattr__(name, value)
```

评论区精华

- gemini-code-assist[bot] 指出融合 kernel 缺少输入验证：建议在 gemma_dual_rmsnorm_residual_scalar 中添加对 x2 和 residual 的形状 / 步幅检查，防止越界。该建议未被代码采纳。
- gemini-code-assist[bot] 指出 resolve_language_model 的 fallback 逻辑问题：最后一行 return model.model 在前置检查都失败时必然引发 AttributeError，建议改用 getattr。该建议未被代码采纳。
- kpham-sgl 要求验证 MMMU 准确率：作者更新 PR body 显示 MMMU 结果 54.89%，匹配官方。
  - 融合 kernel 缺少输入验证 (correctness): 未被代码采纳，风险仍存在。
  - resolve_language_model 的 fallback 引发 AttributeError (correctness): 未被代码采纳，潜在 bug。
  - MMMU 准确率验证 (question): 作者更新 PR body 显示 MMMU 结果 54.89%，匹配官方。

风险与影响

- 风险：
  1. 越界风险：gemma_dual_rmsnorm_residual_scalar 仅检查 x1 的 shape 和 stride，未检查 x2 和 residual，若传入不匹配的张量可能导致 Triton kernel 中的越界访问。
  2. 潜在 AttributeError：resolve_language_model 的最终 fallback return model.model 在模型既无 model 也无 language_model 时必定抛出 AttributeError，虽然当前模型路径可能覆盖，但未来扩展可能触发。
  3. state_dict 污染风险：gemma4_mm.py 中通过 __setattr__ 阻断 model 赋值，但若其他代码直接写入 _modules，可能绕过保护，需保持警惕。
  4. PCG 静默关闭：PCG 启用检查严格依赖属性存在，若模型类命名或属性结构变化，可能静默降级为非 PCG 路径，但已有 warning 日志。 - 影响：直接影响：Gemma4 VLM 部署用户，预填充延迟降低最多 53%，大 token 下降低 5-15%。间接影响：其他 VLM（如 Qwen2.5-VL）不受影响，非 VLM 模型不接触此路径。团队影响：新增了融合 kernel 和 PCG 分支维护成本，但代码量小（+158），结构清晰。 - 风险标记：输入验证缺失或导致越界，PCG 启用条件变更，fallback AttributeError 潜在风险，双分支 kernel 维护成本

关联脉络

- 暂无明显关联 PR