PR #34085 完整报告

sgl-project/sglang

[diffusion] Clean up kernels and shared fast paths

合并时间：2026-08-09 00:37

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34085

执行摘要

- 一句话：集中 diffusion 共享快路径，统一内核与质量门控
- 推荐动作：值得精读。本 PR 展示了扩散内核在性能优化之后的收敛模式：共享数值原语保证 bit-exact 口径、QualityGatedFusion 以 all-or-nothing 方式管理非 bit-exact 融合、模型侧不再各写 fallback。建议重点阅读 quality_gate.py、native_bf16_rmsnorm.py 和 denoising.py 的表驱动挂载逻辑；如果团队要新增扩散模型或新融合内核，这套 " 模型无关命名 + 位精确契约 + 静态守卫 " 的组织方式可以直接借鉴。

功能与动机

PR body 明确指出：近期扩散性能工作在并行推进中留下大量重复，"That left duplicated launch plumbing, repeated numerical helpers, model-local fallback logic, and a few overly narrow or dead components"，并且 "those copies could drift in supported shapes, fallback behavior, or precision contracts"。因此本 PR 的目标是让每个共享关注点只保留一份实现，并使后端选择显式化；lossless 路径继续要求 reference 等价或 bit-exact，非 bit-exact 融合仍限定在 quality=high。

实现拆解

1. 共享数值原语下沉：新增 python/sglang/kernels/ops/diffusion/triton/numerics.py，集中 round_bf16_to_fp32、mul_rn_f32（inline asm 阻断 FMA 收缩）、div_rn_f32、rsqrt_approx_f32、cuda_rsqrtf，让 rmsnorm_scale_shift_bitexact.py、layernorm_modulate.py 等复用，消除各内核之间数值口径漂移的可能。
2. BF16 原生 RMSNorm 通用化：把 Z-Image 模块里重复的 _rmsnorm_scale_kernel、_rmsnorm_tanh_residual_kernel 及启动逻辑抽到新增 native_bf16_rmsnorm.py，公开 rmsnorm_scale、rmsnorm_tanh_residual；同时补强守卫：三者设备一致、全 bf16、shape[-1] <= 8192、非空、weight 形状与连续、行 stride 扁平连续。zimage_native_norm.py 保留 QK RMSNorm 专用实现并同步收紧 dtype/device/numel 守卫。
3. 模型侧快路径入口统一：FLUX、FLUX.2、GLM-Image、ERNIE-Image、LTX-2 各自维护的 _*_residual_gate_add（以及 FLUX 的 _flux_modulate）连同 _DISABLED 全局标志全部删除，改为统一调用 residual_gate_add / modulate_scale_shift，把 dtype 守卫、异常回退、torch.compile 下抛错等策略收口到内核层一处；fused_linear_gelu 的开关判断也从直接读写 _sgl_fused_gelu_enabled 改为 fused_gelu_active()。

4. 质量门控集中管理：新增 quality_gate.py 的 QualityGatedFusion 类（mark/metadata/is_enabled/iter_sites/mount/unmount），denoising 阶段用 _QUALITY_FUSION_HANDLERS 表驱动 _maybe_toggle_quality_fusions，对 linear+GELU、LN+modulate、gate-RMSNorm 三个家族按 batch 的 quality 做 all-or-nothing 挂载 / 卸载。
5. 测试与维护配套：test_zimage_native_norm.py 改名为 test_native_bf16_rmsnorm.py 并扩展拒绝用例；新增 test_quality_gate.py、test_ulysses_qkv.py、test_scale_shift.py；删除针对已移除包装的 test_ernie_residual_gate_add.py；对 fused LN+modulate 增加 torch.compile(fullgraph=True) 覆盖。5 个提交还包括对齐 modulation JIT kernel namespace、修复 quality-gate 测试入口、清理死融合助手等收尾工作。

关键文件：

- python/sglang/multimodal_gen/runtime/models/dits/flux.py（模块 扩散模型；类别 source；类型 data-contract；符号 _flux_residual_gate_add, _flux_modulate, modulate_scale_shift, residual_gate_add）：改动最大的模型文件：删除模型本地 _flux_residual_gate_add 与 _flux_modulate 包装及对应的 _DISABLED 全局标志，统一改调共享入口，并引入 fused_gelu_active 统一判断 GELU 融合开关。
- python/sglang/multimodal_gen/runtime/models/dits/ltx_2.py（模块 扩散模型；类别 source；类型 data-contract；符号 _ltx2_residual_gate_add, residual_gate_add）：LTX-2 旧版 _ltx2_residual_gate_add 没有 half-dtype 限制，统一到共享入口后需确认行为对齐；同时清理了 3 处冗余解包格式化。
- python/sglang/kernels/ops/diffusion/triton/native_bf16_rmsnorm.py（模块 内核层；类别 infra；类型 infrastructure；符号 rmsnorm_scale, rmsnorm_tanh_residual, _rmsnorm_scale_kernel, _rmsnorm_tanh_residual_kernel）：新增的通用 BF16 原生 RMSNorm 融合模块，将 Z-Image 专用的两个 Triton 内核与守卫逻辑抽取为模型无关实现，是本次重构的基础设施核心。
- python/sglang/kernels/ops/diffusion/triton/zimage_native_norm.py（模块 内核层；类别 infra；类型 infrastructure；符号 zimage_qk_rmsnorm_native, can_use_qk_rmsnorm_native）：从 195 行精简到只保留 QK RMSNorm 专用实现，移除重复的通用核心里程碑式收敛；同时加强 QK 路径的 device 一致性与空张量守卫。
- python/sglang/kernels/ops/diffusion/quality_gate.py（模块 门控协议；类别 infra；类型 infrastructure；符号 QualityGatedFusion, mark, is_enabled, iter_sites）：新增 QualityGatedFusion 协议类，统一定义 fusion site 的标记、启用状态与 all-or-nothing 挂载语义，是整个 quality=high 门控机制的核心抽象。
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/denoising.py（模块 去噪管线；类别 source；类型 core-logic；符号 _QUALITY_FUSION_HANDLERS, _maybe_toggle_quality_fusions）：将质量门控挂载逻辑重构为 _QUALITY_FUSION_HANDLERS 表驱动，三个融合家族统一遍历，日志从手写 if/else 收敛为集合驱动输出。
- python/sglang/kernels/ops/diffusion/triton/numerics.py（模块 数值原语；类别 infra；类型 infrastructure；符号 round_bf16_to_fp32, mul_rn_f32, div_rn_f32, rsqrt_approx_f32）：新增共享数值原语模块，用 inline asm 精确控制 fp32 舍入，是多个

bit-exact 内核消除重复实现的关键。

- test/registered/kernels/ops/diffusion/test_native_bf16_rmsnorm.py（模块 测试；类别 test；类型 rename-or-move；符号 test_native_bf16_rmsnorm_rejects_unsupported_inputs, test_rmsnorm_scale_matches_native_bf16, test_rmsnorm_tanh_residual_matches_native_bf16, test_native_bf16_rmsnorm_rejects_hidden_size_above_limit）：由 test_zimage_native_norm.py 改名并扩展，验证通用 RMSNorm 融合与 Z-Image 版本的等价性及拒绝路径，是本次重构的回归防线。

关键符号：rmsnorm_scale, rmsnorm_tanh_residual, QualityGatedFusion.mount, QualityGatedFusion.unmount, QualityGatedFusion.is_enabled, residual_gate_add, modulate_scale_shift, fused_gelu_active, _maybe_toggle_quality_fusions, round_bf16_to_fp32, mul_rn_f32, zimage_qk_rmsnorm_native

关键源码片段

python/sglang/kernels/ops/diffusion/triton/native_bf16_rmsnorm.py

新增的通用 BF16 原生 RMSNorm 融合模块，将 Z-Image 专用的两个 Triton 内核与守卫逻辑抽取为模型无关实现，是本次重构的基础设施核心。

```
# python/sglang/kernels/ops/diffusion/triton/native_bf16_rmsnorm.py
# 从 Z-Image 专用模块抽出的通用 BF16 原生 RMSNorm 融合，供多个扩散模型共享。
import torch
import triton
import triton.language as tl

MAX_HIDDEN_SIZE = 8192


@triton.jit
def _rmsnorm_scale_kernel(
    y_ptr, x_ptr, weight_ptr, scale_ptr,
    x_row_stride, scale_row_stride, seq_len,
    dim: tl.constexpr, eps: tl.constexpr, block_dim: tl.constexpr,
):
    # 每个 program 处理一行；rstd 的求值顺序与 aten bf16 链保持一致，
    # 每次乘 / 规约都落回 bf16，从而与 eager 的两次舍入逐位一致。
    row = tl.program_id(0)
    offsets = tl.arange(0, block_dim)
    mask = offsets < dim

    x = tl.load(x_ptr + row * x_row_stride + offsets, mask=mask, other=0.0)
    square = (x * x).to(tl.bfloat16)
    mean_square = (tl.sum(square, axis=0) / dim).to(tl.bfloat16)
    rstd = tl.rsqrt((mean_square + eps).to(tl.bfloat16).to(tl.float32)).to(tl.bfloat16)

    batch = row // seq_len
    weight = tl.load(weight_ptr + offsets, mask=mask, other=0.0)
```

```
    scale = tl.load(scale_ptr + batch * scale_row_stride + offsets, mask=mask, other=0.0)
    y = (((x * rstd).to(tl.bfloat16) * weight).to(tl.bfloat16) * scale).to(tl.bfloat16)
    tl.store(y_ptr + row * dim + offsets, y, mask=mask)


def _flat_row_stride(x: torch.Tensor) -> int | None:
    # 行 stride 必须严格形成"扁平行"布局，否则拒绝快路径，
    # 避免把非连续视图按连续行处理导致错位。
    if x.dim() < 2 or x.stride(-1) != 1:
        return None
    row_stride = x.stride(-2)
    expected_stride = row_stride * x.shape[-2]
    for dim in range(x.dim() - 3, -1, -1):
        if x.stride(dim) != expected_stride:
            return None
        expected_stride *= x.shape[dim]
    return row_stride


def rmsnorm_scale(
    x: torch.Tensor, weight: torch.Tensor, scale: torch.Tensor, eps: float,
) -> torch.Tensor | None:
    # 统一守卫：同一 CUDA 设备、全部 bf16、隐藏维不超过 8192、
    # 非空输入、weight 形状匹配且连续；任一不满足都返回 None 让调用方回退 eager。
    if not _can_use_operand(x, weight, scale):
        return None

    dim = x.shape[-1]
    x_rows = x.numel() // dim
    scale_rows = scale.numel() // dim
    if x_rows % scale_rows != 0:
        return None

    x_row_stride = _flat_row_stride(x)
    scale_row_stride = _flat_row_stride(scale)
    if x_row_stride is None or scale_row_stride is None:
        return None

    out = torch.empty_like(x, memory_format=torch.contiguous_format)
    with torch.get_device_module().device(x.device):
        _rmsnorm_scale_kernel[(x_rows,)](
            out.reshape(-1, dim),
            x,
            weight,
            scale,
            x_row_stride,
            scale_row_stride,
            x_rows // scale_rows,
            dim,
```

```
            eps,
            block_dim=triton.next_power_of_2(dim),
            num_warps=8,
        )
    return out
```

python/sglang/kernels/ops/diffusion/quality_gate.py

新增 QualityGatedFusion 协议类，统一定义 fusion site 的标记、启用状态与 all-or-nothing 挂载语义，是整个 quality=high 门控机制的核心抽象。

```
# python/sglang/kernels/ops/diffusion/quality_gate.py
class QualityGatedFusion:
    """单个融合家族（family）的挂载协议。

    marker_attr 标记 site，enabled_attr 是普通模块属性，forward 在
    torch.compile 下可直接读取它，而不依赖本 Python 对象。
    """

    __slots__ = ("enabled_attr", "marker_attr", "name")

    def __init__(self, *, name: str, marker_attr: str, enabled_attr: str) -> None:
        self.name = name
        self.marker_attr = marker_attr
        self.enabled_attr = enabled_attr

    def mark(self, module, metadata=True) -> None:
        # 标记 site 并默认关闭，避免未挂载时误走非 bit-exact 路径。
        setattr(module, self.marker_attr, metadata)
        setattr(module, self.enabled_attr, False)

    def is_enabled(self, module) -> bool:
        return bool(getattr(module, self.enabled_attr, False))

    def iter_sites(self, root):
        # marker_attr 是 site 的“身份证”，遍历模块树即可收集整个家族。
        for module in root.modules():
            if hasattr(module, self.marker_attr):
                yield module

    def mount(self, root, *, reject_reason=None, logger=None) -> bool:
        # All-or-nothing：任一 site 不满足静态守卫就整个家族回退 reference，
        # 避免同一家族部分开启导致数值口径不一致。
        sites = list(self.iter_sites(root))
        if not sites:
            return False

        if reject_reason is not None:
            for site in sites:
                reason = reject_reason(site)
```

```
                if reason is None:
                    continue
                self._set_enabled(sites, False)
                if logger is not None:
                    logger.info(
                        "%s: %s site failed static guards (%s); keeping the "
                        "whole model on the reference path",
                        self.name,
                        type(site).__name__,
                        reason,
                    )
                return False

        self._set_enabled(sites, True)
        return True

    def unmount(self, root) -> None:
        self._set_enabled(self.iter_sites(root), False)

    def _set_enabled(self, sites, enabled: bool) -> None:
        for site in sites:
            setattr(site, self.enabled_attr, enabled)
```

python/sglang/multimodal_gen/runtime/pipelines_core/stages/denoising.py

将质量门控挂载逻辑重构为 _QUALITY_FUSION_HANDLERS 表驱动，三个融合家族统一遍历，日志从手写 if/else 收敛为集合驱动输出。

```
# python/sglang/multimodal_gen/runtime/pipelines_core/stages/denoising.py
# 三个 quality=high 融合家族统一注册为处理器表，新增家族只需追加一项。
_QUALITY_FUSION_HANDLERS: tuple[
    tuple[str, Callable[[nn.Module], bool], Callable[[nn.Module], None]], ...
] = (
    (
        "fused linear+GELU (cublasLt epilogue)",
        mount_fused_linear_gelu,
        unmount_fused_linear_gelu,
    ),
    (
        "fused LN+modulate (affine folding)",
        mount_fused_ln_modulate,
        unmount_fused_ln_modulate,
    ),
    (
        "fused gate RMSNorm (BF16-native Triton)",
        mount_fused_gate_rmsnorm,
        unmount_fused_gate_rmsnorm,
    ),
)
```

```
def _maybe_toggle_quality_fusions(self, batch: Req) -> None:
    # quality="high" 挂载全部允许的融合，否则全部卸载；
    # quality 参与动态 batch 签名，因此 batch 内统一，进程级切换是安全的。
    want = getattr(batch.sampling_params, "quality", "lossless") == "high"
    if want == self._quality_fusions_mounted:
        return
    mounted_fusions: set[str] = set()
    for transformer in filter(None, [self.transformer, self.transformer_2]):
        for description, mount, unmount in _QUALITY_FUSION_HANDLERS:
            if want:
                if mount(transformer):
                    mounted_fusions.add(description)
            else:
                unmount(transformer)
    self._quality_fusions_mounted = want
    for description in sorted(mounted_fusions):
        logger.info("Mounted %s for quality=high", description)
```

评论区精华

本 PR 没有来自其他维护者的 review 评论，唯一的评论是作者 BBuf 贴出的 CI 链接。讨论内容以 PR body 中的验证数据为主：H200 上 parent-vs-PR A/B 共 19 个内核用例 × 5 次运行，每次输出 hash 都与父提交一致，中位延迟差异在 -0.90% 到 +0.43% 之间；post-rebase 最终 SHA 上 2700 个相关测试通过、1 个跳过；生产形状 A/B 中 LN+modulate、residual-gate、Ulysses relayout、adaLN modulation 均保持 bit-exact。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 行为对齐差异：统一入口后，原先各模型包装的守卫可能不完全一致。例如 LTX-2 旧版 _ltx2_residual_gate_add 没有 half-dtype 限制，而 FLUX/GLM/ERNIE 旧版都有；若共享 residual_gate_add 以其中一方为准，对另一方是静默行为变化（性能或数值路径）。需确认共享实现覆盖了所有原守卫。
  2. 平台兼容性：numerics.py 中的正确舍入依赖 PTX inline asm（mul.rn.f32、div.rn.f32），在非 NVIDIA 平台（AMD、NPU、Apple Silicon）上不可编译或不可用，依赖调用方的平台 / 设备守卫；H200 上的验证无法覆盖这些平台。
  3. torch.compile 稳定性：QualityGatedFusion 把 enabled 状态放在普通模块属性上以兼容 compile，但如果 site 在编译图捕获后被动态修改，可能造成图不一致；all-or-nothing 的挂载逻辑依赖 batch 内 quality 均匀，改动动态 batch 签名时需要同步审计。
  4. 验证范围：H200 上测试充分，但错误路径、非连续视图、极端 shape 组合只靠单元测试覆盖，跨模型重构下仍有回归风险。 - 影响：影响面覆盖 diffusion 侧全部主要 DiT 模型家族（FLUX、FLUX.2、GLM-Image、ERNIE-Image、LTX-2、Z-Image）的公共快路径：residual-gate、modulate、BF16 RMSNorm、质量门控挂载。对用户无 API 变化

，quality=lossless 保持 bit-for-bit，quality=high 的融合行为由同一套协议统一控制。对团队而言，内核启动约定与 JIT kernel 惯例对齐、模块改为模型无关命名，后续新增扩散模型可直接复用；测试体系从模型本地用例收敛为共享内核用例，降低了维护成本。
- 风险标记：跨模型共享路径重构，位精确性依赖硬件验证，inline PTX asm 平台兼容性，LTX-2 旧路径无 half 守卫，测试聚焦 H200

关联脉络

- PR #34004 [diffusion] FLUX.1 fused adaLN modulate (bit-exact) + RoPE cache hoist, LN-affine folding behind quality=high (H200 e2e -3.5% lossless / -6.9% high): 本 PR 清理的 modulate_scale_shift、residual_gate_add、质量门控正是 34004 引入的融合链路，本次将模型本地包装统一为共享入口。
- PR #34008 [diffusion] GLM-Image bit-exact fused aten LayerNorm+modulate / qk-LN (H200 30-step denoise -8.1%): GLM-Image 的 LN+modulate 融合与新增 numerics / native_bf16_rmsnorm 共享同一套位精确数值约束，本 PR 把其内核原语收敛到共享模块。
- PR #34015 [diffusion] Sana: bit-exact fused aten LayerNorm+modulate under BCG (H200 denoise -4.8%): 同属 diffusion 融合内核加速系列，本 PR 的共享数值原语与 JIT 启动约定可直接被 Sana 类内核复用。
- PR #33400 [jit_kernel] Move JIT kernels into namespace sglang: 本 PR 提交 "Align modulation JIT kernel namespace" 与 33400 的 namespace 统一工作呼应，延续了 JIT 内核组织规范的收敛。
- PR #34106 [jit_kernel] Fix missing JIT kernel namespaces: 后续修复 JIT kernel namespace 缺失的 PR，与本 PR 涉及的 JIT 内核命名空间对齐属于同一演进脉络。