PR #34172 完整报告

sgl-project/sglang

[diffusion] LTX-2 quality=high fused RMSNorm+modulate + FFN GELU epilogue (H200 ltx23-one-stage denoise 45.85->43.24 s, ~matches torch.compile)

合并时间：2026-08-10 09:46

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34172

执行摘要

- 一句话：LTX-2 新增 quality=high 融合核，去噪提速 5.7% 逼近 torch.compile
- 推荐动作：值得精读，三个看点：（1）profiling 归因方法——用 kernel-time 归因定位 torch.compile 优势全部来自 elementwise 融合，从而避免盲目上 CUDA graph，方法论可直接复用于其他 compute-bound 模型；（2）quality 门控的精度工程——显式承认 ≤ 1 ULP 差异，用质量档位而非运行时自检管理精度边界，lossless 默认保持 bit-exact；（3）复用与缓存——ones 权重缓存让 weightless RMSNorm 直接复用既有 fused_rmsnorm_scale_shift_bitexact 内核。建议后续将 PSNR 验收固化为 e2e CI 测试，并为 QualityGatedFusion handler 家族补充文档。

功能与动机

PR body 的 profiling 结论是方案起点：LTX-2 视频去噪在 H200 eager 下 GPU 占用约 89%，属计算密集型，可拆分的 CUDA graph 几乎无收益；torch.compile 的优势几乎全部来自两个 elementwise 链的融合——每个 transformer block 的 adaLN RMSNorm(x) * (1 + scale) + shift 调制，以及 FFN 的 GELU。kernel-time 归因显示 compile 把 GPU-op 总时间从 8.58M µs 压到 7.37M µs（-14%），GEMM 与 attention 均未改动。因此本 PR 的目标是在 eager 里手工复现这两处融合，让 high 档位逼近 compile，同时避开 compile 分钟级 warmup 与跨会话不确定性。

实现拆解

1. Profiling 归因确定切入点：在 H200 上对比 ltx23-one-stage 的 5 步去噪 eager 与 torch.compile，确认 compute-bound（GPU-busy ≈ 89%）、CUDA graph 无收益、compile 的 -14% GPU-op 时间全部来自 elementwise 融合（aten vectorized_layer_norm、通用 elementwise、GELU 启动折叠为 Triton 融合核），GEMM/attention 未动。这决定了方案不做 CUDA graph，而是复现两处融合。
2. 新增内核封装模块：新建 python/sglang/kernels/ops/diffusion/ltx2_rmsnorm_modulate.py，复用 QualityGatedFusion 基座（与 fused_linear_gelu、fused_ln_modulate、fused_gate_rmsnorm 同族），提供 mark_ltx2_rms_norm_modulate_site、mount_ltx2_rms_norm_modulate、unmount_ltx2_rms_norm_modulate、ltx2_rms_norm_modulate_active 四个钩子；利用 LTX-2 的 RMSNormNoWeight 无权重特性，通过按 (device, hidden) 缓存的 ones 张量复现均方根归一化语义；直接复用现有

fused_rmsnorm_scale_shift_bitexact Triton 内核，将 rms_norm(x) * (1 + scale) +shift 折叠为一次启动。can_fuse_ltx2_rms_norm_modulate 守卫仅放行 CUDA bf16，其余回退 eager。

3. 模型接入：修改 python/sglang/multimodal_gen/runtime/models/dits/ltx_2.py，LTX2TransformerBlock.forward 中 6 个 adaLN 位置（视频 / 音频自注意力、视频 / 音频 prompt 交叉注意力、视频 / 音频 FFN 前调制）统一经新增 _ltx2_rms_norm_modulate 路由；LTX2FeedForward 在 __init__ 中执行 mark_fused_gelu_site(self, "proj_in")，forward 在 fused_gelu_active 且 can_fuse_linear_gelu 通过时走 fused_linear_gelu_tanh（cublasLt GELU epilogue），未挂载时保持原 proj_in + GELU 路径。
4. 管线注册：在 python/sglang/multimodal_gen/runtime/pipelines_core/stages/denoising.py 的 _QUALITY_FUSION_HANDLERS 元组新增 "LTX-2 fused RMSNorm+modulate" 成员，使 quality="high" 请求在批次边界统一 mount/unmount，与既有三个融合族（linear+GELU、LN+modulate、gate RMSNorm）共存；未标记站点的模型不受影响。
5. 测试与验收：新增 test/registered/kernels/ops/diffusion/test_ltx2_rms_norm_modulate.py（注册 CUDA base-b-kernel-unit 与 AMD nightly CI），覆盖未挂载路径 torch.equal 于 eager 参考、挂载路径精确等于融合核输出且相对 eager 保持在 bf16 半精度舍入内；手工完成 PSNR 验收（全 121 帧 min 35.29 / mean 36.34 dB，验收线 > 25 dB）与 H200 性能基准（one-stage denoise 45.85 → 43.24 s）。

关键文件：

- python/sglang/multimodal_gen/runtime/models/dits/ltx_2.py（模块 模型层；类别 source；类型 core-logic；符号 _ltx2_rms_norm_modulate, LTX2FeedForward.forward, LTX2TransformerBlock.forward）：模型主路径：6 个 adaLN 位置与 FFN GELU 全部接入融合路由，是本 PR 的业务落地点，也是回归风险集中处（+69/-17）。
- python/sglang/kernels/ops/diffusion/ltx2_rmsnorm_modulate.py（模块 融合内核；类别 infra；类型 infrastructure；符号 mark_ltx2_rms_norm_modulate_site, ltx2_rms_norm_modulate_active, mount_ltx2_rms_norm_modulate, unmount_ltx2_rms_norm_modulate）：新增内核封装模块：复用 QualityGatedFusion 与既有 Triton 内核，引入 ones 权重缓存，是 quality=high 融合能力的核心载体。
- test/registered/kernels/ops/diffusion/test_ltx2_rms_norm_modulate.py（模块 单元测试；类别 test；类型 test-coverage；符号 _setup, _eager, _inputs, test_lossless_default_is_bitexact）：新增测试覆盖两条关键行为：lossless 默认逐位一致、high 挂载走融合核且精度在 bf16 舍入内，并注册 CUDA/AMD CI。
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/denoising.py（模块 去噪管线；类别 source；类型 dependency-wiring）：在 quality fusion handler 家族中注册新成员，决定 quality=high 请求的挂载时机，是本 PR 功能生效的开关。

关键符号：_ltx2_rms_norm_modulate, fused_ltx2_rms_norm_modulate, can_fuse_ltx2_rms_norm_modulate, _ones_weight, mark_ltx2_rms_norm_modulate_site, mount_ltx2_rms_norm_modulate, unmount_ltx2_rms_norm_modulate, ltx2_rms_norm_modulate_active, LTX2FeedForward.forward, LTX2TransformerBlock.forward

关键源码片段

python/sglang/multimodal_gen/runtime/models/dits/ltx_2.py

模型主路径：6 个 adaLN 位置与 FFN GELU 全部接入融合路由，是本 PR 的业务落地点，也是回归风险集中处（+69/-17）。

```
# ---- python/sglang/multimodal_gen/runtime/models/dits/ltx_2.py（重构后）----

def _ltx2_rms_norm_modulate(
    block: nn.Module,
    rms_norm: nn.Module,
    x: torch.Tensor,
    scale: torch.Tensor,
    shift: torch.Tensor,
    eps: float,
) -> torch.Tensor:
    # 挂载且 per-call 守卫通过时走融合内核（≤ 1 bf16 ULP 差异）
    # 否则执行原样 eager 链，保证 lossless 默认路径逐位不变
    if ltx2_rms_norm_modulate_active(block) and can_fuse_ltx2_rms_norm_modulate(
        x, scale, shift
    ):
        return fused_ltx2_rms_norm_modulate(x, scale, shift, eps)
    return rms_norm(x, eps) * (1 + scale) + shift

# LTX2TransformerBlock.forward 中视频自注意力支路的 adaLN 位置
# 原为内联表达式 self.rms_norm(...) * (1 + vscale_msa) + vshift_msa
norm_hidden_states = _ltx2_rms_norm_modulate(
    self, self.rms_norm, hidden_states, vscale_msa, vshift_msa, self.norm_eps
)

# LTX2FeedForward.forward：proj_in 与 tanh-GELU 融合为 cublasLt epilogue
# fused_gelu_active / can_fuse_linear_gelu 双守卫，未挂载时保持原路径
def forward(self, x: torch.Tensor) -> torch.Tensor:
    if fused_gelu_active(self) and can_fuse_linear_gelu(self.proj_in, x):
        x = fused_linear_gelu_tanh(x, self.proj_in.weight, self.proj_in.bias)
    else:
        x, _ = self.proj_in(x)
        x = self.act(x)
    x, _ = self.proj_out(x)
    return x
```

python/sglang/kernels/ops/diffusion/ltx2_rmsnorm_modulate.py

新增内核封装模块：复用 QualityGatedFusion 与既有 Triton 内核，引入 ones 权重缓存，是 quality=high 融合能力的核心载体。

```
# ---- python/sglang/kernels/ops/diffusion/ltx2_rmsnorm_modulate.py（新增）----

_SITE_MARKER_ATTR = "_sgl_ltx2_rms_norm_modulate_site"
_SITE_ENABLED_ATTR = "_sgl_ltx2_rms_norm_modulate_enabled"
```

```
# 与 fused_linear_gelu / fused_gate_rmsnorm 同族的 QualityGatedFusion 基座：
# mark 声明融合站点，mount/unmount 在 quality="high" 请求的批次边界切换
_FUSION = QualityGatedFusion(
    name="LTX-2 RMSNorm+modulate",
    marker_attr=_SITE_MARKER_ATTR,
    enabled_attr=_SITE_ENABLED_ATTR,
)

# RMSNormNoWeight 不施加权重，ones 向量即可精确复现其数值语义；
# 按 (device, hidden) 缓存，避免每个 batch 重复分配 ones 张量
_ONES_WEIGHT_CACHE: dict[tuple[torch.device, int], torch.Tensor] = {}


def _ones_weight(x: torch.Tensor) -> torch.Tensor:
    key = (x.device, int(x.shape[-1]))
    w = _ONES_WEIGHT_CACHE.get(key)
    if w is None:
        w = torch.ones(x.shape[-1], device=x.device, dtype=torch.bfloat16)
        _ONES_WEIGHT_CACHE[key] = w
    return w


def can_fuse_ltx2_rms_norm_modulate(
    x: torch.Tensor, scale: torch.Tensor, shift: torch.Tensor
) -> bool:
    # 仅 CUDA bf16 允许融合；其余情况回退 eager 参考路径
    if x.dtype is not torch.bfloat16 or not x.is_cuda:
        return False
    # 复用既有 Triton 内核的布局 / 规格守卫
    return can_use_fused_rmsnorm_scale_shift(x, _ones_weight(x), scale, shift)


def fused_ltx2_rms_norm_modulate(
    x: torch.Tensor, scale: torch.Tensor, shift: torch.Tensor, eps: float
) -> torch.Tensor:
    # 单次内核启动完成 rms_norm(x) * (1 + scale) + shift
    return fused_rmsnorm_scale_shift_bitexact(x, _ones_weight(x), scale, shift, eps)
```

test/registered/kernels/ops/diffusion/test_ltx2_rms_norm_modulate.py

新增测试覆盖两条关键行为：lossless 默认逐位一致、high 挂载走融合核且精度在 bf16 舍入内，并注册 CUDA/AMD CI。

```
# ---- test/registered/kernels/ops/diffusion/test_ltx2_rms_norm_modulate.py（新增）----

# hidden 4096 = LTX-2 视频流，2048 = 音频流
@pytest.mark.parametrize("hidden", [4096, 2048])
def test_lossless_default_is_bitexact(hidden):
    # 标记但未挂载的站点（lossless 默认）必须与 eager 参考逐位一致
    block = nn.Module()
```

```
    mark_ltx2_rms_norm_modulate_site(block)
    rms, x, scale, shift = _inputs(hidden)
    out = _ltx2_rms_norm_modulate(block, rms, x, scale, shift, 1e-6)
    assert torch.equal(out, _eager(rms, x, scale, shift, 1e-6))


@pytest.mark.parametrize("hidden", [4096, 2048])
def test_mounted_high_uses_fused_kernel(hidden):
    block = nn.Module()
    mark_ltx2_rms_norm_modulate_site(block)
    assert mount_ltx2_rms_norm_modulate(block)
    try:
        rms, x, scale, shift = _inputs(hidden)
        out = _ltx2_rms_norm_modulate(block, rms, x, scale, shift, 1e-6)
        # 挂载路径必须精确等于融合内核输出
        assert torch.equal(out, fused_ltx2_rms_norm_modulate(x, scale, shift, 1e-6))
        # 且相对 eager 参考保持在 bf16 半精度舍入范围内
        ref = _eager(rms, x, scale, shift, 1e-6)
        assert torch.allclose(out.float(), ref.float(), atol=3e-2, rtol=1e-2)
    finally:
        unmount_ltx2_rms_norm_modulate(block)
```

评论区精华

本 PR 没有任何 review 评论（review_comments_count = 0），由作者 BBuf 自审并直接合并，缺乏独立审核交峰。唯一留下的讨论是 issue 评论中的 CI 状态说明：

> "CI green on all required checks after re-running two flaky lanes: 14/14 call-multimodal-gen-tests SUCCESS (the initial jit-kernel-b200-test / multimodal-gen-test-1-5090 reds cleared on rerun with no code change), finish/lint/gate SUCCESS. Remaining reds are the repo's known non-required AMD/NPU/finish-aggregator lanes (red on merged #34008/#34085 as well)."

两条初始失败的 lane 经重跑转绿且无代码变更，剩余红色为仓库已知的非必需 AMD/NPU lane，与本次变更无关。

- CI 状态与 flaky lane 重跑 (other)：两条 flaky lane 经重跑通过，代码无需修改；非必需 lane 的红色与本次变更无关。

风险与影响

- 风险：
  1. 精度路径：融合核非 bit-exact（rsqrt.approx vs aten 的 rsqrtf，≤ 1 bf16 ULP），仅 quality="high" 启用；PSNR 验证（min 35.29 dB）通过，但该验证是手工 benchmark，未固化为自动化 CI 回归测试，后续内核微调可能引入精度漂移而不被及时发现。
  2. 默认路径安全：lossless 默认完全未改动，仍走原样 eager 参考链（bit-exact），默认用户无回归风险。

3. 模型主路径改动：ltx_2.py 的 forward 修改了 6 个 adaLN 调用点，若 can_fuse_ltx2_rms_norm_modulate 或 mount 门控出现漏洞（如非 CUDA / 非 bf16 环境误启用），会产生静默数值偏差；当前有双重守卫（mount 状态 + per-call 检查）与单元测试兜底。
4. 平台依赖：融合路径依赖 fused_rmsnorm_scale_shift_bitexact Triton 内核在目标平台的可用性，AMD 等平台由 can_use_fused_rmsnorm_scale_shift 判定，不可用即回退 eager，性能收益随之消失但正确性不受影响。
5. 审核缺口：该 PR 无 review 评论即合并，融合路径（尤其精度门控语义）缺少独立技术审核，属于流程性风险。 - 影响：用户侧：LTX-2 推理用户在 quality="high" 下获得约 5.7% 去噪提速（one-stage 场景），与 torch.compile 相当但无 warmup；quality 语义新增中间档位，lossless 默认行为逐位不变。系统侧：新增 python/sglang/kernels/ops/diffusion/ltx2_rmsnorm_modulate.py 模块，denoising 阶段 fusion handler 增至 4 族；其他 Diffusion 模型未标记站点，完全不受影响。团队侧：进一步确立 "QualityGatedFusion + 站点标记 + 批次边界挂载 " 的可扩展融合模式，后续模型接入成本低，但该模式目前缺少成文文档，依赖代码内注释传递约定。 - 风险标记：非 bit-exact 精度路径（≤ 1 bf16 ULP），仅 quality=high 启用，默认 lossless 不变，PSNR 验收未固化为自动化 CI，依赖既有 Triton 内核可用性，无 review 审核直接合并

关联脉络

- PR #34174 [diffusion] BCG: auto-capture the default warmup resolution instead of hard-requiring --warmup-resolutions (H200 SANA denoise 0.73->0.457 s with a single flag): 同属 multimodal_gen diffusion 性能优化主线，BCG 与内核融合是两条互补的 eager 加速路径，均在近期合入（34174 早于本 PR）。
- PR #33702 [diffusion] Add Sol-Attn sparse attention backend for diffusion: 同为 diffusion 后端性能优化家族（稀疏注意力提速），与本次 elementwise 融合共同构成 diffusion runtime 的加速矩阵。
- PR #33471 runtime: Add flashinfer rmsnorm + quant fusion support SM90, SM100, SM120- #32994: 同为 RMSNorm 类内核融合思路（RMSNorm+FP8 量化融合），本 PR 复用的 fused_rmsnorm_scale_shift 属于同一融合内核家族。
- PR #34186 [CI] Key scheduled CUDA suites by runner_config instead of hand-written jobs: 本 PR 新增测试使用 register_cuda_ci/register_amd_ci 注册式 CI，与 CI 调度迁移（runner_config 注册制）直接衔接。