PR #34125 完整报告

sgl-project/sglang

[diffusion] Bit-exact data-movement elimination for the Wan causal VAE decoder (H200 LongLive2 704x1280x61f: decode 2.80->2.32 s lossless / 2.12->1.67 s quality=high, e2e -10.7%)

合并时间：2026-08-09 09:50

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34125

执行摘要

- 一句话：Wan VAE 解码器融合数据搬运内核，lossless 解码提速 17%
- 推荐动作：值得精读。本 PR 是 kernel-level 数据搬运消除的典范：以 strict bit-exact（非近似）为约束，用融合内核替代多次全张量遍历，并以多层次验证（kernel 级 bitwise 单测 + 端到端 md5 + 与既有 quality gate 组合验证）支撑"纯数据搬运"论点。值得关注的设计决策包括：紧凑缓存与 conv 输入同一 pass 双输出（实现缓存簿记完全消除）、输出布局严格跟随 aten（保护下游 layout-sensitive 逻辑）、完备的 fallback 条件与可交错性。对 diffusion/ 视频推理、Triton 内核开发、性能工程团队均有参考价值。

功能与动机

在少步数 Wan 家族视频管线中，因果 VAE 解码是固定成本大头：LongLive2（704x1280、61 帧、4 步 DMD、H200 bf16）解码阶段占端到端 62%，FastWan2.2-TI2V-5B（fp32 解码）占 81%。内核级归因显示 2.59 s 解码 GPU 时间里卷积本身仅约 1.26 s，0.51 s 花在 aten::copy_（1849 次调用）、60 ms fill_、39 ms cat 等围绕因果特征缓存与 channels_last_3d 布局的簿记操作上，另有约 0.28 s 消耗在 DupUp3D 捷径分支。PR body 明确目标是'移除解码器在默认 lossless 路径上的数据搬运税'，且强调所有改动都是纯数据移动加零填充，因此不需要质量门控，golden 输出经 md5 端到端验证不变。

实现拆解

本 PR 的实现分为三个层次：

1. 新增 Triton 数据搬运内核模块（python/sglang/kernels/ops/diffusion/triton/wan_causal_cache.py）：
   - _cat_pad_cl3d_kernel / cat_pad_channels_last_3d：以一次 kernel launch 完成 contiguous_cl3d(F.pad(cat([cache, x], dim=2), padding))，通过 strided 读取（读 x 与 cache 各自 stride）、NDHWC 写出、kernel 内零填充；keep_cache_t > 0 时同一遍额外写出下一 chunk 的紧凑特征缓存（未填充的内部末尾帧），返回 (conv_input, cache) 二元组，从而完全消除逐 chunk 的 clone/cat 簿记，且 conv 输入缓冲在卷积后即可释放。
   - _dup_up3d_add_kernel / dup_up3d_add：一次 gather+add 完成 main + DupUp3D(src)，输出用 empty_like(main)（与 aten add 输出布局一致），遍历顺序跟

随输出内存序（NHWC 风格 arm 通道最内层）保证存储与 main 加载连续；pixel-shuffle 因子作为 constexpr 2 的幂，除法编译为移位 / 掩码。

2. 重接线 wanvae.py 的六个特征缓存卷积点（python/sglang/multimodal_gen/runtime/models/vaes/wanvae.py）：
   - 新增 _cache_payload（取出缓存条目的 Tensor 载荷，区分空槽与 "Rep" 标记）、_fused_conv_cache_supported（严格检查是否 CUDA、WanCausalConv3d 类型、5 维输入、AMP 支持、权重 channels_last_3d、非 torch.compiler.is_compiling()）、_run_cached_causal_conv（统一调度 fused 快速路径与原始 aten 回退路径）。
   - 六个卷积点（residual block 的 conv1/conv2、编码器 / 解码器的 conv_in/conv_out、WanResample 的 time_conv，含 "Rep" 首 chunk 标记）全部改为经 _run_cached_causal_conv 路由；WanCausalConv3d.forward 在无缓存有 padding 场景走 fused builder；residual_up_block_forward 的 main + avg_shortcut(x_copy) 在捷径为 DupUp3D 且满足条件时走 dup_up3d_add。
   - 所有快速路径均保留完整回退：非 CUDA、spatial-parallel conv 子类、权重非 NDHWC、torch.compile 追踪、设备 /dtype 不匹配、非对称 padding 等一律落到逐位一致的原始 aten 链；紧凑缓存持有与参考缓存完全相同的值，因此 fused 与回退 chunk 可自由交错。
3. 测试与 CI 配套（test/registered/kernels/ops/diffusion/test_wan_causal_cache.py）：
   - 新增注册 CUDA 单测（register_cuda_ci(est_time=40, stage="base-b-kernel-unit", runner_config="1-gpu-large")），覆盖 test_cat_pad_bitwise（dtype/shape/ 带 stride 缓存视图 / 首 chunk 零填充 /legacy 1 帧缓存 / 编码器 T=4/ 仅时间轴 padding/ 双输出缓存发射）、test_dup_up3d_add_bitwise（含 first_chunk 切片、输出 stride 与 aten add 一致）、test_cached_conv_chunk_loop_bitwise（从 None 与 "Rep" 起始的逐 chunk 卷积循环，与强制回退路径逐块 bitwise 对比）。
   - 端到端验证：LongLive2（bf16）与 FastWan2.2-TI2V-5B（fp32）各 10/10 请求帧流 md5 与 main 完全一致；--quality high 输出与 main 的 --quality high 字节一致，证明与 #33546 数值上可组合。

关键文件：

- python/sglang/multimodal_gen/runtime/models/vaes/wanvae.py（模块 VAE 模型；类别 source；类型 core-logic；符号 _cache_payload, _fused_conv_cache_supported, _run_cached_causal_conv）：核心改造文件：六个特征缓存卷积点与 DupUp3D 捷径分支统一接入 fused 路径，新增 _cache_payload/_fused_conv_cache_supported/_run_cached_causal_conv 三个关键符号，并保留完整的逐位一致 aten 回退链。
- python/sglang/kernels/ops/diffusion/triton/wan_causal_cache.py（模块 Triton 内核；类别 infra；类型 core-logic；符号 _cat_pad_cl3d_kernel, cat_pad_channels_last_3d, _dup_up3d_add_kernel, dup_up3d_add）：新增 383 行 Triton 内核模块，是性能收益的来源：cat_pad_channels_last_3d 单遍构建 NDHWC conv 输入并双输出紧凑缓存，dup_up3d_add 单遍 gather+add 完成捷径分支。
- test/registered/kernels/ops/diffusion/test_wan_causal_cache.py（模块 单元测试；类别 test；类型 test-coverage；符号 _cl3d, _ref_cat_pad, test_cat_pad_bitwise, test_dup_up3d_add_bitwise）：新增注册 CUDA 单测（164 行），以 bitwise 相等断言覆盖两个内核与 aten 链的一致性，以及从 None/"Rep" 起始的逐 chunk 缓存交错正确性，是

bit-exact 主张的直接证据。

关键符号：_run_cached_causal_conv, _fused_conv_cache_supported, _cache_payload, cat_pad_channels_last_3d, dup_up3d_add, _cat_pad_cl3d_kernel, _dup_up3d_add_kernel

关键源码片段

python/sglang/multimodal_gen/runtime/models/vaes/wanvae.py

核心改造文件：六个特征缓存卷积点与 DupUp3D 捷径分支统一接入 fused 路径，新增 _cache_payload/_fused_conv_cache_supported/_run_cached_causal_conv 三个关键符号，并保留完整的逐位一致 aten 回退链。

```
# python/sglang/multimodal_gen/runtime/models/vaes/wanvae.py
# 新增的 fused 路径调度核心：先做严格的支持性检查，再尝试 fused 内核，
# 任何不满足条件时都回退到原始 aten 链（逐位一致）。

def _fused_conv_cache_supported(conv: nn.Module, x: torch.Tensor) -> bool:
    return (
        cat_pad_channels_last_3d is not None # Triton 内核可导入
        and type(conv) is WanCausalConv3d # 精确类型，排除并行卷积子类
        and x.dim() == 5
        and x.is_cuda
        and current_platform.is_amp_supported()
        and _conv3d_weight_is_channels_last_3d(conv.weight)
        and not torch.compiler.is_compiling() # torch.compile 追踪时禁用
    )


def _run_cached_causal_conv(
    conv: nn.Module,
    x: torch.Tensor,
    cache_list: list,
    idx: int,
) -> torch.Tensor:
    """运行一次因果卷积，同时消费并刷新其特征缓存槽。

    快速路径（与 aten 链逐位一致，纯数据搬运加零填充）：用一个内核直接把
    conv 输入（缓存帧 + 隐状态 + padding）构建为 channels_last_3d，并把下
    一 chunk 的缓存条目作为该输入未填充尾部的紧凑拷贝一并产出，替代逐
    chunk 的 clone/cat 簿记。fused 与回退 chunk 可自由交错。
    """
    cache = cache_list[idx]
    is_rep = isinstance(cache, str) # "Rep" 标记来自 WanResample
    payload = None if is_rep else _cache_payload(cache)
    if _fused_conv_cache_supported(conv, x) and (
        payload is None or (payload.device == x.device and payload.dtype == x.dtype)
    ):
        # 同一 kernel pass 同时产出 conv 输入与紧凑缓存，
```

```
            # 这样 conv 输入缓冲在卷积结束后即可释放，不会被缓存钉住。
            pair = cat_pad_channels_last_3d(x, payload, conv._padding, keep_cache_t=CACHE_T)
            if pair is not None:
                inp, cache_list[idx] = pair
                return nn.Conv3d.forward(conv, inp)
        # 原始 aten 路径（逐位一致的簿记）。
        cache_x = x[:, :, -CACHE_T:, :, :].clone()
        if cache_x.shape[2] < 2 and payload is not None:
            # 缓存上一个 chunk 的最后一帧。
            cache_x = torch.cat(
                [payload[:, :, -1, :, :].unsqueeze(2).to(cache_x.device), cache_x],
                dim=2,
            )
        elif cache_x.shape[2] < 2 and is_rep:
            cache_x = torch.cat(
                [torch.zeros_like(cache_x).to(cache_x.device), cache_x],
                dim=2,
            )
        out = conv(x) if payload is None else conv(x, payload)
        cache_list[idx] = cache_x
        return out
```

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

新增 383 行 Triton 内核模块，是性能收益的来源：cat_pad_channels_last_3d 单遍构建 NDHWC conv 输入并双输出紧凑缓存，dup_up3d_add 单遍 gather+add 完成捷径分支。

```
# python/sglang/kernels/ops/diffusion/triton/wan_causal_cache.py
# 单遍构建因果卷积输入的 Triton 内核：替换 cat + F.pad + contiguous 三次
# 全张量遍历。输出按 channels_last_3d 连续索引展开，x 与 cache 均为 strided
# 读取；padding 区域与越界位置直接零填充，等价于 aten 的 F.pad 行为。

@triton.jit
def _cat_pad_cl3d_kernel(
    x_ptr, cache_ptr, out_ptr, keep_ptr, total, C, T, H, W,
    cache_t, out_t, out_h, out_w, pad_t_zero, pad_h, pad_w,
    sxb, sxc, sxt, sxh, sxw, scb, scc, sct, sch, scw,
    HAS_CACHE: tl.constexpr, KEEP_T: tl.constexpr,
    IDX64: tl.constexpr, BLOCK: tl.constexpr,
):
    offs = tl.program_id(0) * BLOCK + tl.arange(0, BLOCK)
    mask = offs < total

    # channels_last_3d 线性索引 = (((b*T+t)*H+h)*W+w)*C + c
    oc = offs % C
    rest = offs // C
    ow = rest % out_w
    rest = rest // out_w
    oh = rest % out_h
    rest = rest // out_h
```

```
    o_t = rest % out_t
    ob = rest // out_t

    iw = ow - pad_w
    ih = oh - pad_h
    it = o_t - pad_t_zero

    spatial_ok = (iw >= 0) & (iw < W) & (ih >= 0) & (ih < H)
    from_cache = spatial_ok & (it >= 0) & (it < cache_t)
    from_x = spatial_ok & (it >= cache_t) & (it < cache_t + T)

    xt = it - cache_t
    x_off = ob * sxb + oc * sxc + xt * sxt + ih * sxh + iw * sxw
    vals = tl.load(x_ptr + x_off, mask=mask & from_x, other=0.0)
    if HAS_CACHE:
        c_off = ob * scb + oc * scc + it * sct + ih * sch + iw * scw
        c_vals = tl.load(cache_ptr + c_off, mask=mask & from_cache, other=0.0)
        vals = tl.where(from_cache, c_vals, vals)
    tl.store(out_ptr + offs, vals, mask=mask)

    if KEEP_T > 0:
        # 同一遍附带产出紧凑缓存：未填充内部最后 KEEP_T 帧，
        # 布局 channels_last_3d，形状 (B, C, KEEP_T, H, W)。
        ct = o_t - (out_t - KEEP_T)
        keep = mask & spatial_ok & (ct >= 0)
        k_off = (((ob * KEEP_T + ct) * H + ih) * W + iw) * C + oc
        tl.store(keep_ptr + k_off, vals, mask=keep)
```

评论区精华

本 PR 的 review 评论很少（comments_count=2，均为作者 BBuf 自述），核心讨论集中在两点：

- Rebase 后 lossless 字节一致性复确认：作者在 PR 中评论确认 rebase 到当前 main 后 LongLive2（704x1280x61f、4 步 DMD、seed 42、quality=lossless、H200）有无本 PR 输出相同帧 md5 93468c316ad55b4aea26903629d98b91（3/3 请求），且内核级 bitwise 一致性由注册单测覆盖（22 个测试通过）。
- CI 偶发性能阈值抖动与归属澄清：初始 multimodal-gen-test-1-gpu (2) shard 变红，作者说明是延迟断言超阈值 1-6% 的瞬时抖动（如 693 vs 684 ms），涉及 joyai_image_edit / flux_2 / Wan InputValidationStage，与本次 VAE 解码路径无关；同一 shard 在 #34126 同池通过后重跑变绿。正确性由字节级 md5 与 22 个 bitwise 单测独立证明。剩余的 *-amd/*-npu/pr-test-*-finish 红为已知非必需通道（#34008 / #34085 同样红）。
  - Rebase 后 lossless 字节一致性确认 (correctness): 确认 lossless 路径在 rebase 后仍字节级一致，正确性主张成立。
  - CI 偶发失败的性质判断 (testing): 判定为瞬时 perf-threshold flake，非本 PR 引入；正确性由 md5 与 bitwise 单测独立证明。

风险与影响

- 风险：
  1. bit-exact 主张依赖严格前置条件：fused 路径的启用条件是 _fused_conv_cache_supported 的长度检查链（CUDA、WanCausalConv3d 精确类型、5 维、AMP、权重 channels_last_3d、非编译追踪），任何条件不满足即回退 aten 链。风险在于未来新增的 VAE 用法（如非对称 padding、多帧 chunk、torch.compile 场景）若未被 cat_pad_channels_last_3d 的返回 None 分支覆盖，可能静默回退而非报错——这是有意的设计，但需要测试持续兜底。
  2. 布局敏感的下游依赖：dup_up3d_add 明确保证输出 stride 与 aten add 完全一致（测试断言 out.stride() == ref.stride()），因为后续 layout-sensitive reduction 依赖内存格式；若未来上游 main 的布局假设变化，该内核需要同步调整。
  3. 边界数值条件：cat_pad_channels_last_3d 对 total > _MAX_INT32 * 4 返回 None（回退），pad_t_zero < 0（缓存片数超过前填充）同样回退；这些是防御性边界，但超大张量场景（如更高分辨率长视频）下的实际性能收益需要另行确认。
  4. CI 性能阈值脆弱性：PR 提及的瞬时 perf-threshold flake 表明相关 multimodal-gen 测试的延迟断言存在抖动，虽与本 PR 无关，但提示该测试体系对同池 runner 波动敏感。
- 影响：
  1. 用户可感知的性能提升：Wan 家族视频管线（LongLive2、FastWan2.2-TI2V-5B 等共享同一 VAE）的默认 lossless 解码路径无需任何配置即可提速：LongLive2 DecodingStage -17.1%、端到端 -10.7%，峰值显存从 49.6 GB 降至 46.1 GB（-3.5 GB）；FastWan2.2-TI2V-5B（fp32 解码）DecodingStage -14.8%、端到端 -12.0%。叠加 --quality high（#33546）后整体解码可较当前默认再降 40%。
  2. 系统层面：每个 chunk 一次 pass 替代四次全张量遍历，显著降低显存带宽压力与临时缓冲占用（conv 输入缓冲不再被缓存持有），对多请求并发（显存敏感）有利。
  3. 团队 / 维护层面：六个重复的 cache 簿记片段收敛为一个 _run_cached_causal_conv 辅助函数，后续新增卷积点只需调用该 helper；fused 内核与回退路径可交错，降低行为漂移风险。新增 383 行 Triton 内核与 164 行单测，属于可维护的增量。 - 风险标记：核心路径变更，数值一致性断言强依赖测试覆盖，新增 Triton 内核仅 CUDA 路径，CI 性能阈值抖动

关联脉络

- PR #33546 Quality-gated Wan RMSNorm+SiLU fast path: 本 PR body 明确提及并依赖 #33546 的 --quality high 档位；两个 PR 作用于同一 Wan VAE 解码链的不同阶段（#33546 加速 pointwise norm 链，本 PR 消除默认 lossless 路径的数据搬运税），且经 md5 验证可数值组合。
- PR #34126 [diffusion] FLUX.1: route the adaLN LN+modulate sites through the bit-exact fused LayerNorm+modulate kernel: 同一批次的 diffusion 性能优化 PR，采用相同的 bit-exact 融合内核方法论（输出与 aten 链逐位一致），且 CI 讨论中 H200 同 runner 池互通，体现当前 diffusion 性能工程的统一方向。

- PR #34015 [diffusion] Sana: bit-exact fused aten LayerNorm+modulate under BCG: 同类 bit-exact 融合内核优化（Sana 的 adaLN 融合），与 #34126、#33546 共同构成 diffusion 模型全家桶的 kernel 融合演进趋势。