PR #29281 完整报告

sgl-project/sglang

[KDA-Pilot] Add diffusion causal Conv3D cat-pad CUDA fast path for Cosmos3

合并时间：2026-06-26 15:06

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/29281

执行摘要

- 一句话：为 Cosmos3 添加 CUDA cat/pad 快速路径，提速 ~2x
- 推荐动作：值得精读。重点关注 causal_conv3d_cat_pad.py 中的自定义 op 注册与 fake_impl 设计，以及 parallel_conv.py 中的 fallback 模式。该 PR 展示了 JIT 内核与 torch.compile 兼容的典型集成方案，对后续扩散模型优化有参考价值。

功能与动机

Cosmos3-Nano T2V trace 显示 VAE decode 中共 48 次 cat+pad 调用（8 种 shape），原有 Triton 内核成为瓶颈。参考 KDA-Pilot 任务 #132 开发 CUDA 快速路径，目标 kernel 级 2x 加速，降低 VAE decode 延迟。

实现拆解

1. CUDA 内核开发（causal_conv3d_cat_pad.cuh）：基于 MIT HAN Lab Kernel Design Agents 开发，使用 flat-chunk 16-byte vectorized store，通过 flat 索引反向分解输出坐标，一次性完成 cat + pad + zero-fill 操作。
2. Python JIT 封装（causal_conv3d_cat_pad.py）：利用 cache_once + load_jit 加载内核；定义 fake_impl 供 torch.compile 进行形状推导；通过 register_custom_op 注册为不透明自定义操作，避免 trace 时触发 JIT 加载。
3. 模型层集成与 fallback（parallel_conv.py）：新增 fused_causal_conv3d_cat_pad 函数，优先尝试 CUDA 快速路径，若 JIT 加载或运行失败则记录 warning 并回退到 Triton；使用全局哨兵变量避免重复失败开销。
4. torch.compile 适配（cosmos3.py）：将 gen_layers 编译参数从 dynamic=True 改为 dynamic=False，与自定义 op 的静态形状需求一致。
5. 测试与基准（test_causal_conv3d_cat_pad.py, bench_causal_conv3d_cat_pad.py）：覆盖 Cosmos3 全部 8 种 shape 的 bitwise 精度验证，包含 torch.compile(fullgraph=True) 测试；基准测试提供 Triton vs CUDA 逐 shape 性能对比。

关键文件：

- python/sglang/jit_kernel/diffusion/causal_conv3d_cat_pad.py（模块 JIT 内核；类别 source；类型 core-logic；符号 _jit_causal_conv3d_cat_pad_module, _causal_conv3d_cat_pad_fake_impl, _causal_conv3d_cat_pad_custom_op, fused_causal_conv3d_cat_pad_cuda）：核心 Python 封装：JIT 加载、fake_impl、自定

义 op 注册、can_use 检查。是整个 CUDA 快速路径的总入口。

- python/sglang/multimodal_gen/runtime/layers/parallel_conv.py（模块 模型层；类别 source；类型 dependency-wiring；符号 fused_causal_conv3d_cat_pad）：模型集成入口：修改 fused_causal_conv3d_cat_pad 函数，添加 CUDA 快速路径的首选尝试与 Triton 回退逻辑，以及异常日志。
- test/registered/jit/diffusion/test_causal_conv3d_cat_pad.py（模块 测试；类别 test；类型 test-coverage；符号 _make_inputs, test_causal_conv3d_cat_pad, test_causal_conv3d_cat_pad_torch_compile, fn）：单元测试：覆盖 Cosmos3 全部 8 种 shape 的正确性，以及 torch.compile(fullgraph=True) 路径。
- test/registered/jit/benchmark/diffusion/bench_causal_conv3d_cat_pad.py（模块 基准；类别 test；类型 test-coverage；符号 Case, make_inputs, benchmark）：基准测试：提供 Triton vs CUDA 的性能对比，便于后续回归检查。
- python/sglang/jit_kernel/csrc/diffusion/causal_conv3d_cat_pad.cuh（模块 CUDA 内核；类别 source；类型 core-logic；符号 cat_pad_flat_kernel, CausalConv3dCatPadKernel）：CUDA 内核实现：flat-chunk 16-byte vectorized store，通过 flat 索引反向分解输出坐标，一次性完成 cat+pad+zero-fill 操作。
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/cosmos3.py（模块 扩散管线；类别 source；类型 configuration）：torch.compile 配置调整：将 dynamic=True 改为 False，确保自定义 op 在静态形状下可被 torch.compile 正确 tracing。

关键符号：_jit_causal_conv3d_cat_pad_module, _causal_conv3d_cat_pad_fake_impl, _causal_conv3d_cat_pad_custom_op, fused_causal_conv3d_cat_pad_cuda, can_use_fused_causal_conv3d_cat_pad_cuda, fused_causal_conv3d_cat_pad, cat_pad_flat_kernel

关键源码片段

python/sglang/multimodal_gen/runtime/layers/parallel_conv.py

模型集成入口：修改 fused_causal_conv3d_cat_pad 函数，添加 CUDA 快速路径的首选尝试与 Triton 回退逻辑，以及异常日志。

```
from sglang.multimodal_gen.runtime.utils.logging_utils import init_logger

logger = init_logger(__name__)

# 全局哨兵，避免反复尝试失败的 CUDA 路径
_causal_conv3d_cat_pad_cuda_failed = False


def fused_causal_conv3d_cat_pad(
    x: torch.Tensor,
    cache_x: torch.Tensor,
    padding: list[int],
) -> torch.Tensor:
```

```
    global _causal_conv3d_cat_pad_cuda_failed
    # 条件：CUDA 实现可用、can_use 通过、且之前未失败
    if (
        fused_causal_conv3d_cat_pad_cuda is not None
        and can_use_fused_causal_conv3d_cat_pad_cuda(x, cache_x, padding)
        and not _causal_conv3d_cat_pad_cuda_failed
    ):
        try:
            return fused_causal_conv3d_cat_pad_cuda(x, cache_x, padding)
        except Exception:
            # 记录失败信息，帮助开发者诊断 JIT 问题（如编译错误、不支持的架构）
            logger.warning(
                "fused_causal_conv3d_cat_pad_cuda failed, falling back to Triton",
                exc_info=True,
            )
            _causal_conv3d_cat_pad_cuda_failed = True
    if fused_causal_conv3d_cat_pad_triton is None:
        raise RuntimeError("causal Conv3D cat/pad fusion is only available on CUDA")
    return fused_causal_conv3d_cat_pad_triton(x, cache_x, padding)
```

评论区精华

- CUDA 内核循环优化建议：gemini-code-assist 建议在 cat_pad_flat_kernel 循环的最后一次迭代中避免冗余索引更新和指针重算（if (i < kVec - 1)）。该建议未采纳，因为内核为 memory-bound，影响微小。
- 异常日志记录建议：gemini-code-assist 建议在 CUDA JIT 失败时记录异常详情。该建议已被采纳，作者在 commit 91b3662 中增加了 logger.warning 并附带 exc_info=True。
  - CUDA 内核循环最后迭代冗余操作优化 (performance): 该建议未在最终代码中采纳，因为影响微小且内核为 memory-bound。
  - CUDA JIT 失败时应记录异常日志 (other): 作者在后续 commit (91b3662) 中采纳，增加了 logger.warning 和 exc_info=True。

风险与影响

- 风险：
  - JIT 编译兼容性：不同 CUDA 版本或 GPU 架构可能导致内核编译失败，但 fallback 到 Triton 可保证功能正确，仅损失性能。
  - 对齐精度假设：can_use 函数要求输出总元素数 16-byte 对齐（out_numel % vec_elems == 0），若不对齐则无法利用向量化存储，会回退到 Triton；但未对齐场景下 Triton 路径仍然正确。
  - 测试覆盖局限：单元测试仅覆盖 Cosmos3 的 8 种生产 shape，未覆盖非对齐、非连续或 padding 组合不同的边缘场景。
  - 自定义 op 形状一致性：fake_impl 的输出形状必须与 CUDA 内核实际写入的形状完全一致，否则 torch.compile 会生成错误代码。当前实现基于固定 padding 模式（pad_d_right=0），若引入新 padding 组合需同步更新。

- 影响：
  - 用户影响：Cosmos3 模型用户无需任何代码修改即可获得约 2% E2E 加速（VAE *decode* 阶段），且无精度损失。
  - 系统影响：新增约 600 行代码（CUDA 内核 255 行，Python 封装 141 行，测试 181 行，集成修改 41 行）；SGLang JIT 内核框架新增一种可复用的自定义 op 集成范例。
  - 团队影响：展示了如何将外部开发（KDA-Pilot）的 CUDA 内核快速集成到 SGLang 并在 torch.compile 生态下安全使用，为后续类似贡献提供了参考模式。
  - 风险标记：JIT 编译兼容性，对齐精度假设，测试覆盖仅限 Cosmos3 形状

关联脉络

- 暂无明显关联 PR