PR #27081 完整报告

sgl-project/sglang

[diffusion] Use Conv2d width padding in WanVAE

合并时间：2026-06-03 10:14

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27081

执行摘要

- 一句话：WanVAE 使用 Conv2d 原生宽度 padding
- 推荐动作：可精读，作为如何利用框架原生特性替代手动 pad 的案例。性能提升有限，但代码简洁性提升明显。

功能与动机

减少显存分配和 kernel 启动开销，通过利用 Conv2d 原生 padding 机制优化性能。PR body 提到 *decode* 阶段耗时从 5.2121s 降至 5.1284s（~1.5%），峰值显存从 44174MB 降至 43522MB（~1.5%）。

实现拆解

1. 修改 WanDistConv2d 初始化中的 padding 分配逻辑：在 python/sglang/multimodal_gen/runtime/models/vaes/parallel/wan_dist_utils.py 的 __init__ 中，当 height_halo_size > 0 时，_padding 从 (padding[1], padding[1], 0, 0) 改为 (0, 0, 0, 0)，即不再为宽度方向保留 padding；同时将 self.padding 从 (0, 0) 改为 (0, self.padding[1])，将宽度方向的 padding 交由 nn.Conv2d 基类处理。当 height_halo_size == 0 时，_padding 的宽度部分也置零，高度部分保持不变。
2. 优化 forward 中的 F.pad 调用：在 forward 中，用 if any(self._padding): x = F.pad(x, self._padding) 替代了无条件的 x = F.pad(x, self._padding)。由于大部分情况下 _padding 为全零，跳过了无意义的 pad 操作。
3. 验证正向一致性：输出 byte-identical（MP4 sha256 一致），确保功能无退化。

关键文件：

- python/sglang/multimodal_gen/runtime/models/vaes/parallel/wan_dist_utils.py（模块 扩散模型；类别 source；类型 data-contract；符号 WanDistConv2d.init, WanDistConv2d.forward）：唯一修改的文件，核心变更位于 WanDistConv2d 类的 __init__和 forward 方法，将宽度 padding 从显式 F.pad 迁移到 Conv2d 原生 padding。

关键符号：WanDistConv2d.init, WanDistConv2d.forward

关键源码片段

python/sglang/multimodal_gen/runtime/models/vaes/parallel/wan_dist_utils.py

唯一修改的文件，核心变更位于 WanDistConv2d 类的 __init__和 forward 方法，将宽度 padding 从显式 F.pad 迁移到 Conv2d 原生 padding。

```
# python/sglang/multimodal_gen/runtime/models/vaes/parallel/wan_dist_utils.py
# 本 PR 将宽度方向 padding 从显式 F.pad 迁移到 Conv2d 原生 padding，
# 减少一次显存分配和 kernel 启动，提升 decode 阶段性能约 1.5%。

class WanDistConv2d(nn.Conv2d):
    def __init__(...):
        # ... 原有初始化逻辑 ...

        # 旧代码：当 height_halo_size > 0 时，
        # self._padding = (self.padding[1], self.padding[1], 0, 0)
        # 新代码：置零宽度 padding，交由 Conv2d 基类处理
        if self.height_halo_size > 0:
            self._padding = (0, 0, 0, 0)
        else:
            self._padding = (
                0,
                0,
                self.padding[0],
                self.padding[0],
            )

        # 旧代码：self.padding = (0, 0) # 完全禁用 Conv2d 原生 padding
        # 新代码：保留宽度方向的 padding，让 Conv2d 负责宽度填充
        self.padding = (0, self.padding[1])

    def forward(self, x):
        # 旧代码：无条件调用 F.pad
        # x = F.pad(x, self._padding)
        # 新代码：仅当 _padding 非全零时调用（大多数情况可跳过）
        if any(self._padding):
            x = F.pad(x, self._padding)

        x_padded, ... = halo_exchange(x, ...)
        # ... 后续 forward 逻辑不变 ...
        out = super().forward(x_padded)
        return out
```

评论区精华

无 review 讨论，PR 由作者自行合并。

- 暂无高价值评论线程

风险与影响

- 风险：回归风险：虽然显式保证了 byte-identical，但若未来修改 Conv2d 的 padding 行为（例如换后端），宽度 padding 可能被重复应用（Conv2d 的 padding 和 _padding 中的宽度 padding）导致错误。当前 PR 通过将 _padding 宽度部分置零消除了此风险。 性能：warmup 后无需额外 kernel，正向性能提升稳定。 兼容性：仅影响 WanDistConv2d 内部逻辑，对外接口和高度 padding 语义未变。
- 影响：影响范围：仅修改 WanDistConv2d 类，影响所有使用 WanVAE 的模型（包括 Cosmos3 等）的 decode 阶段。性能提升约 1-2%，显存降低约 1-2%。影响程度：低，属于微优化。
- 风险标记：微优化，高度特定于 WanVAE

关联脉络

- PR #27077 [diffusion] Preserve dtype in WanVAE nearest upsample: 同样优化 WanVAE 性能，且修改的文件同属 wan_common_utils.py 和 wan_dist_utils.py 涉及的 WanVAE 底层操作。
- PR #27086 [diffusion] Clamp WanVAE decode output in place: 同为 WanVAE decode 阶段性能优化，修改了 wanvae.py。
- PR #27084 [diffusion] Optimize Cosmos3 i2v latent prep: 同为 diffusion pipeline 性能优化，影响 Cosmos3 模型，与本 PR 在性能改进方向一致。