# PR #49030 完整报告

vllm-project/vllm

[Bugfix][Multimodal] Fix video temporal padding estimates

合并时间：2026-07-29 01:57

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/49030

## 执行摘要

- 一句话：修复视频帧数填充公式，支持 temporal_patch_size > 2
- 推荐动作：建议尽快合并，作为最小修复方案。值得注意的设计决策是选择不引入共享辅助函数（如 round_up），直接在六个调用点复制修改，避免了对 math_utils 等公共模块的依赖，降低了重构风险。但长期看，可考虑未来统一抽象。

## 功能与动机

Issue #47866 指出当 temporal_patch_size > 2 时，旧公式 num_frames + num_frames % temporal_patch_size 不能将帧数正确填充到下一个倍数，例如 17 % 4 = 18 而不是 20，导致 grid_t 和估计的视频 token 数偏小，影响 vLLM 的预算估计、dummy input 尺寸和预热步骤。PR body 也详细解释了这一问题。

## 实现拆解

1. 修复核心公式：在六个模型的 _get_vision_info 方法中，将填充公式从 num_frames + num_frames % temporal_patch_size 改为 num_frames + (-num_frames % temporal_patch_size)，确保帧数总能向上取整到 temporal_patch_size 的整数倍。同时更新注释引用指向 Transformers v5.13.0 的视频处理器实现。
2. 影响文件：修改了 vllm/model_executor/models/qwen2_vl.py、glm4_1v.py、kanana_v.py、keye.py、llava_onevision2.py、mimo_v2_omni.py 六个文件，每个文件只改动一行公式和一行注释（MiMo 额外涉及类型转换以适应其 effective_frames 计算）。
3. 添加回归测试：在 tests/models/multimodal/processing/test_glm4_1v.py 中新增 test_vision_info_rounds_up_temporal_frames 测试用例，使用 Mock 对象固定 17 帧，参数化 temporal_patch_size 为 2、4、8，验证 grid_t 分别为 9、5、3，覆盖核心修复逻辑。

关键文件：

- tests/models/multimodal/processing/test_glm4_1v.py（模块 测试；类别 test；类型 test-coverage；符号 test_vision_info_rounds_up_temporal_frames）：新增回归测试，验证不同 temporal_patch_size 下的 token 估计正确性，是变更的核心验证。
- vllm/model_executor/models/glm4_1v.py（模块 视频处理；类别 source；类型 data-contract）：主要修改的模型之一，展示了公式变更和注释更新。
- vllm/model_executor/models/kanana_v.py（模块 视频处理；类别 source；类型 data-contract）：第三个展示相同公式修复的模型文件。

- vllm/model_executor/models/qwen2_vl.py（模块 视频处理；类别 source；类型 data-contract）：核心模型之一，公式修复的起源模型。
- vllm/model_executor/models/keye.py（模块 视频处理；类别 source；类型 data-contract）：受影响模型之一，仅一行公式变更。
- vllm/model_executor/models/llava_onevision2.py（模块 视频处理；类别 source；类型 data-contract）：受影响模型之一，仅一行公式变更。
- vllm/model_executor/models/mimo_v2_omni.py（模块 视频处理；类别 source；类型 data-contract）：受影响模型之一，额外需要将填充后的帧数转换为 int。

关键符号：_get_vision_info, test_vision_info_rounds_up_temporal_frames

## 关键源码片段

### tests/models/multimodal/processing/test_glm4_1v.py

新增回归测试，验证不同 temporal_patch_size 下的 token 估计正确性，是变更的核心验证。

```
@pytest.mark.parametrize(
    ("temporal_patch_size", "expected_grid_t"),
    [(2, 9), (4, 5), (8, 3)],
)
def test_vision_info_rounds_up_temporal_frames(
    temporal_patch_size: int,
    expected_grid_t: int,
):
    # 创建 Mock 对象模拟 Glm4vProcessingInfo
    info = Mock(spec=Glm4vProcessingInfo)
    vision_config = info.get_hf_config.return_value.vision_config
    vision_config.patch_size = 14
    vision_config.spatial_merge_size = 2
    vision_config.temporal_patch_size = temporal_patch_size

    # 调用 _get_vision_info 获取 vision token 数（传入固定 17 帧）
    _, num_vision_tokens = Glm4vProcessingInfo._get_vision_info(
        info,
        image_width=28,
        image_height=28,
        num_frames=17,
        do_resize=False,
    )

    # 断言 token 数与预期 grid_t 一致，确保填充正确
    assert num_vision_tokens == expected_grid_t
```

### vllm/model_executor/models/glm4_1v.py

主要修改的模型之一，展示了公式变更和注释更新。

```
# 计算填充后的帧数，确保能被 temporal_patch_size 整除
# 参考 Transformers v5.13.0 视频处理器实现：
```

```
# https://github.com/huggingface/transformers/blob/v5.13.0/src/transformers/models/qwen2_vl/video_processing_qwen2_vl.py#L249-L252
padded_num_frames = num_frames + (-num_frames % temporal_patch_size)

grid_t = max(padded_num_frames // temporal_patch_size, 1)
grid_h = preprocessed_size.height // patch_size
grid_w = preprocessed_size.width // patch_size

num_patches = grid_t * grid_h * grid_w
num_vision_tokens = num_patches // (merge_size**2)

return preprocessed_size, num_vision_tokens
```

## 评论区精华

审查者 Isotr0py 在 glm4_1v.py 的第 1086 行提出疑问，指出视频加载器中已经有帧填充逻辑（vllm/multimodal/video.py 中的代码）。作者 labAxiaoming 回复解释了区别：_get_vision_info 是 vLLM 侧的 token 预算估计、dummy input 尺寸和预热路径，并非实际视频预处理路径；实际输入仍由 HF/model 的图片或视频处理器处理。加载器中的填充只保证帧数为偶数，而 temporal_patch_size 可能为 4、8 等，因此仍需在估计时正确填充。审查者对解释表示认可，最终批准了 PR。

- 视频加载器已有填充与估计路径区别 (question): 审查者认可解释，PR 被批准。

## 风险与影响

- 风险：风险极低。仅修改了六个模型文件中的一行计算公式，且该公式只用于 vLLM 内部的 token 预算估计，不影响实际的图像 / 视频预处理、模型权重或推理输出。新公式与 Transformers 官方实现一致，并且测试覆盖了不同 temporal_patch_size 组合。回归风险主要可能来自其他未更新的模型（如果有类似的复制粘贴错误），但 PR 已覆盖已知的六个模型。MiMo-V2-Omni 额外需要将填充后的帧数转换为 int，但 effective_frames 本身是整数，改动安全。
- 影响：影响范围限定在使用了 temporal_patch_size > 2 的视频输入场景。修复后，vLLM 的 token 预算将更准确，避免了潜在的 grid_t 低估导致的预热不足或内存分配错误。对用户透明，输出结果不变。团队通过添加回归测试确保了后续维护质量。
- 风险标记：低风险，公式修正，复制粘贴遗留

## 关联脉络

- PR #47876 Fix video temporal padding token estimates: 草案 PR，包含类似修复但引入了共享辅助函数；本 PR 是有意的最小替代方案，仅修改调用点。
- PR #47866 [Bug][Multi-modal]: Video frames should be paded right by temporal_patch_size: 关联 Issue，报告了相同问题，PR 修复该 Issue。