PR #44509 完整报告

vllm-project/vllm

[Bugfix] MiniCPM-V-4.6 video inference crash: placeholder count mismatches visual embedding count

合并时间：2026-06-04 23:22

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/44509

执行摘要

- 一句话：修复 MiniCPM-V-4.6 视频推理崩溃
- 推荐动作：值得精读，这是一个典型的 bugfix PR，展示了多模态 pipeline 中数据流不一致的排查与修复思路。设计决策（优先使用已处理尺寸、在数据流源头记录实际尺寸）具有通用借鉴意义。建议合并后为 MiniCPM-V-4.6 添加视频回归测试。

功能与动机

发送视频请求到 openbmb/MiniCPM-V-4_6 会导致 EngineDeadError，因为 placeholder 数量与视觉塔产生的嵌入数量不匹配。图像推理正常，仅视频触发崩溃。这不是回退——视频路径在 vLLM 上从未对 MiniCPM-V-4.6 正常工作。

实现拆解

1. 修复 get_frame_size 中的 HWC/CHW 检测（vllm/multimodal/parse.py）：在 VideoProcessorItems.get_frame_size 中，当 image 是 np.ndarray 或 torch.Tensor 时，检查其维度：若为 3 维且最后一维在 (1, 3, 4) 中，则视为 HWC 格式，此时从 shape[0] 和 shape[1] 读取高和宽；否则视为 CHW 格式，从 shape[1] 和 shape[2] 读取。
2. 记录每帧尺寸并输出 video_image_sizes（vllm/model_executor/models/minicpmv4_6.py）：在 process_videos 中，遍历帧时根据帧类型（PIL Image、numpy 数组或 torch Tensor）获取实际宽高，并以 [W, H] 形式存入 frame_sizes 列表，处理完一个视频后 torch.stack 得到 per_video_image_sizes。最后在返回的字典中添加 "video_image_sizes": per_video_image_sizes。
3. 优先使用已处理的帧尺寸计算占位符（vllm/model_executor/models/minicpmv4_6.py）：在 get_video_replacement 中，从 out_mm_kwargs["video"] 获取 video_image_sizes 数据，若存在则直接使用其第一帧的尺寸和帧数来确定 num_placeholders，避免回退到可能错误的 VideoProcessorItems.get_frame_size 路径。

关键文件：

- vllm/model_executor/models/minicpmv4_6.py（模块 模型执行；类别 source；类型 core-logic；符号 MiniCPMV4_6MultiModalProcessor.process_videos, MiniCPMV4_6MultiModalProcessor._get_prompt_updates, MiniCPMV4_6MultiModalProcessor.get_video_replacement）：核心修复文件：在

process_videos 中收集每帧实际尺寸并返回 video_image_sizes；在 get_video_replacement 中优先使用该字段计算占位符数量。导入新增 numpy、PIL.Image、ImageSize。

- vllm/multimodal/parse.py（模块 多模态解析；类别 source；类型 core-logic；符号 VideoProcessorItems.get_frame_size）：辅助修复文件：在 VideoProcessorItems.get_frame_size 中增加 HWC 格式检测，避免因错误假设 CHW 导致帧尺寸错误，确保回退路径也返回正确尺寸。

关键符号：MiniCPMV4_6MultiModalProcessor.process_videos, MiniCPMV4_6MultiModalProcessor._get_prompt_updates, MiniCPMV4_6MultiModalProcessor.get_video_replacement, VideoProcessorItems.get_frame_size

关键源码片段

vllm/model_executor/models/minicpmv4_6.py

核心修复文件：在 process_videos 中收集每帧实际尺寸并返回 video_image_sizes；在 get_video_replacement 中优先使用该字段计算占位符数量。导入新增 numpy、PIL.Image、ImageSize。

```
# vllm/model_executor/models/minicpmv4_6.py
# 在 process_videos 中，每帧收集实际尺寸 (W, H) 并返回 video_image_sizes
def process_videos(self, mm_data, mm_kwargs, tok_kwargs):
    ...
    per_video_pixel_values: list[torch.Tensor] = []
    per_video_tgt_sizes: list[torch.Tensor] = []
    per_video_image_sizes: list[torch.Tensor] = [] # 新增：记录每帧尺寸

    for video in parsed_videos:
        all_slices: list[torch.Tensor] = []
        ts_list: list[torch.Tensor] = []
        frame_sizes: list[torch.Tensor] = []
        for frame in video:
            # 根据帧类型获取实际宽高，支持 PIL、numpy 和 torch
            if isinstance(frame, PILImage.Image):
                w, h = frame.size
            elif isinstance(frame, np.ndarray):
                if frame.ndim == 3 and frame.shape[-1] in (1, 3, 4):
                    # HWC 格式（常见于 np.array(PIL.Image) 转换结果）
                    h, w = frame.shape[0], frame.shape[1]
                else:
                    # CHW 格式
                    _, h, w = frame.shape
            elif isinstance(frame, torch.Tensor):
                if frame.ndim == 3 and frame.shape[-1] in (1, 3, 4):
                    h, w = frame.shape[0], frame.shape[1]
                else:
                    _, h, w = frame.shape
```

```
            else:
                raise TypeError(f"Unsupported frame type: {type(frame)}")
            frame_sizes.append(torch.tensor([w, h], dtype=torch.long, device="cpu"))
            ...
        per_video_image_sizes.append(torch.stack(frame_sizes))
    return {
        "video_pixel_values": per_video_pixel_values,
        "video_tgt_sizes": per_video_tgt_sizes,
        "video_image_sizes": per_video_image_sizes, # 新增输出
    }

# 在 get_video_replacement 中优先使用已处理的帧尺寸计算占位符
def get_video_replacement(item_idx: int):
    video_mm_kwargs = out_mm_kwargs.get("video")
    if video_mm_kwargs is not None and item_idx < len(video_mm_kwargs):
        video_item = video_mm_kwargs[item_idx]
        image_sizes_elem = video_item.get("video_image_sizes")
        if image_sizes_elem is not None and image_sizes_elem.data is not None:
            # image_sizes_elem.data: (num_frames, 2) – each row is [W, H]
            image_sizes = image_sizes_elem.data
            num_frames = image_sizes.shape[0]
            frame_size = ImageSize(
                width=int(image_sizes[0, 0].item()),
                height=int(image_sizes[0, 1].item()),
            )
            # 使用第一帧尺寸和帧数计算占位符数量，与视觉编码器保持一致
            ...
    # 回退到旧路径（从 VideoProcessorItems 获取尺寸）
    ...
```

vllm/multimodal/parse.py

辅助修复文件：在 VideoProcessorItems.get_frame_size 中增加 HWC 格式检测，避免因错误假设 CHW 导致帧尺寸错误，确保回退路径也返回正确尺寸。

```
# vllm/multimodal/parse.py
class VideoProcessorItems(ProcessorBatchItems[HfVideoItem | None]):
    ...
    def get_frame_size(self, item_idx: int) -> ImageSize:
        ...
        image = video[0]
        if isinstance(image, PILImage.Image):
            return ImageSize(*image.size)
        if isinstance(image, (np.ndarray, torch.Tensor)):
            # 检测 HWC 格式：3 维且最后一维为 1、3 或 4
            if image.ndim == 3 and image.shape[-1] in (1, 3, 4):
                # HWC（例如 np.array(PIL.Image) 的结果，通道在最后一维）
                h, w = image.shape[0], image.shape[1]
            else:
                # CHW（标准 PyTorch/numpy 约定）
```

```
            _, h, w = image.shape
        return ImageSize(w, h)
    assert_never(image)
```

评论区精华

无实质性 review 讨论。仅 bot 自动评论了预检查失败，提交者 tc-mb 随后更新修复了预检查问题，并请求合并。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 回归风险低：改动集中在新添加的 video_image_sizes 字段和帧尺寸检测逻辑，不影响现有图像路径和已有的 video_pixel_values、video_tgt_sizes 输出；旧路径 (VideoProcessorItems.get_frame_size) 仅修复了 HWC 识别，未改变 CHW 行为。
  2. 兼容性：新增的 video_image_sizes 字段不影响现有调用方，因为返回字典可以包含未使用的键；仅 get_video_replacement 会尝试读取它，若不存在则回退旧路径。
  3. 测试覆盖：未添加新的单元测试或集成测试，仅作者进行了手动验证（ tensor-parallel-size 2，NVIDIA 4090）。建议上游添加视频测试用例以确保未来变更不破坏此修复。
- 影响：
  - 用户影响：MiniCPM-V-4.6 视频推理从完全崩溃变为正常工作，直接解锁该模型在 vLLM 上的视频处理能力。
  - 系统影响：无系统级风险，改动量小（2 文件，+60/-2 行）。
  - 团队影响：维护者需关注类似多模态模型（如 MiniCPM-V 其他版本）是否也存在帧尺寸解析问题。
  - 风险标记：缺少测试覆盖

关联脉络

- 暂无明显关联 PR