

PR #53808 完整报告

vllm-project/vllm

[Bugfix][Multimodal] Honor modality-scoped mm_processor_kwargs

合并时间: 2026-08-31 14:37

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/53808>

执行摘要

- 一句话: 修复模态级 mm_processor_kwargs 不生效与视频 / 图像预算串扰
- 推荐动作: 值得精读。核心价值在于 overlay_modality_mm_kwargs 的接口设计 (modality=None 保旧行为、非 Mapping 忽略、scoped 优先于 flat、保留嵌套键) 以及“读取侧按模态展开、HF 侧保留嵌套”的双轨策略; 对维护多模态模型的工程师有直接参考价值。建议后续为 GLM4 / Eagle 等其他模型按同一模式接入, 并在文档中明确 scoped 与 flat 同时出现时的优先级。

功能与动机

Issue #52834 指出两条失败路径: scoped 形式 (videos_kwargs) 被 vLLM 自身读取点静默忽略, 导致编码器缓存按约 12K 视频 token 预留、运行时却可能产生约 229K token 项, 属于 #51603 描述的 encoder cache starvation / livelock 故障类; flat 形式 (size) 则把图像最大项从约 16K 抬到约 448K token, 默认 processor cache 下直接中止启动 (#52835)。issue 作者强调 HF processor 本身接受模态级嵌套形式 (Qwen3VLProcessor._merge_kwargs 会 honor videos_kwargs), vLLM 应与 HF 语义对齐; 此前唯一 workaround 是手工改 checkpoint 内的 video_preprocessor_config.json, 不可从 serve 配置复现。

实现拆解

变更入口是 `vllm/multimodal/processing/context.py` 的

`InputProcessingContext.get_merged_mm_kwargs`, 它被所有模型的多模态读取点共用; 随后在 Qwen2/3-VL 读取点逐个接入。实现分 5 步:

1. 新增模态映射与叠加函数: 在 `context.py` 定义 `_HF_MODALITY_PROCESSOR_KWARGS` (image→images_kwargs、video→videos_kwargs、audio→audio_kwargs) 与 `overlay_modality_mm_kwargs`。函数返回 dict 副本: modality=None 时原样复制保持旧行为; 只叠加指定模态的嵌套 dict; 非 Mapping 值直接忽略; scoped 键优先于同名扁平键, 且保留嵌套键本身供 HF 侧继续读取。
2. 扩展合并入口: `get_merged_mm_kwargs` 增加 keyword-only 参数 `modality`, 先走原有 `merge_mm_processor_kwargs` 合并再 `overlay`。call_hf_processor 与 processor 构造路径不传 `modality`, 保证嵌套 kwargs 原样到达 HF processor。

3. Qwen3-VL 读取点接入 (vllm/model_executor/models/qwen3_vl.py) : `_get_vision_info` 新增 `modality` 参数, 未指定时按 `processor` 类型推断 (Qwen3VLVideoProcessor → video, 否则 image) ; `get_max_video_tokens`、`get_dummy_mm_data`、`_apply_hf_processor_main` 的视频逐项 `size` 拷贝显式传 `modality="video"`。
4. Qwen2-VL 读取点接入 (vllm/model_executor/models/qwen2_vl.py) : `_get_vision_info` 同样加 `modality`; `get_num_image_tokens` 传 "image"、`get_num_video_tokens` 传 "video"、`get_image_size_with_most_features` 默认分支传 `modality="image"`。
5. 测试与类型配套: `tests/multimodal/test_processing.py` 新增 5 个 `overlay` 单测; `tests/models/multimodal/processing/test_qwen3_vl.py` 新增 `_probe_mm_token_budgets` 与回归测试, 复刻 issue 的 CPU 探测脚本。第二个 commit 把返回类型从 `dict[str, object]` 放宽为 `dict[str, Any]`, 因为收紧的类型让 GLM4 / Eagle 等调用点的 `int()` 与 `**kwargs` 被 `mypy` 拒绝。

关键文件:

- `vllm/multimodal/processing/context.py` (模块 多模态处理; 类别 source; 类型 core-logic ; 符号 `overlay_modality_mm_kwargs`, `get_merged_mm_kwargs`) : 核心改动: 新增 `overlay_modality_mm_kwargs` 与 `_HF_MODALITY_PROCESSOR_KWARGS` 映射, `get_merged_mm_kwargs` 增加模态参数, 是全部行为的枢纽。
- `vllm/model_executor/models/qwen3_vl.py` (模块 模型实现; 类别 source; 类型 data-contract; 符号 `_get_vision_info`, `get_max_video_tokens`, `get_dummy_mm_data`, `_apply_hf_processor_main`) : 修复目标模型: token 预算、dummy 输入、逐项视频 `size` 拷贝共 4 处读取点显式按 video 模态读取, `_get_vision_info` 支持按 `processor` 类型推断模态。
- `vllm/model_executor/models/qwen2_vl.py` (模块 模型实现; 类别 source; 类型 data-contract; 符号 `_get_vision_info`, `get_num_image_tokens`, `get_num_video_tokens`, `get_image_size_with_most_features`) : 同源模型同步接入: `_get_vision_info` 支持模态参数, `get_num_image_tokens` / `get_num_video_tokens` 显式区分模态, 图像默认预算读取走 image 作用域。
- `tests/multimodal/test_processing.py` (模块 单元测试; 类别 test; 类型 test-coverage; 符号 `test_overlay_modality_mm_kwargs_scoped_video_does_not_leak_to_image`, `test_overlay_modality_mm_kwargs_flat_size_stays_shared`, `test_overlay_modality_mm_kwargs_scoped_wins_over_flat_for_modality`, `test_overlay_modality_mm_kwargs_ignores_non_mapping_scoped_value`) : `overlay` 函数行为单测, 覆盖 `scoped` 不泄漏、`flat` 保持共享、`scoped` 优先、非 Mapping 忽略与 `merge` 后叠加保序。
- `tests/models/multimodal/processing/test_qwen3_vl.py` (模块 回归测试; 类别 test; 类型 test-coverage; 符号 `_probe_mm_token_budgets`, `test_processor_kwargs_videos_kwargs_does_not_leak_into_image_budget`) : 端到端回归: 用 issue 的 CPU 探测脚本验证 `videos_kwargs` 只抬视频预算、`flat` 仍双模态共享, 防止 #52834 复现。

关键符号: `overlay_modality_mm_kwargs`, `get_merged_mm_kwargs`, `_get_vision_info`, `get_max_video_tokens`, `get_dummy_mm_data`, `_apply_hf_processor_main`,

get_num_image_tokens, get_num_video_tokens, get_image_size_with_most_features

关键源码片段

vllm/multimodal/processing/context.py

核心改动：新增 overlay_modality_mm_kwargs 与 _HF_MODALITY_PROCESSOR_KWARGS 映射，get_merged_mm_kwargs 增加模态参数，是全部行为的枢纽。

```
# HuggingFace processors 除了共享的扁平命名空间，还接受嵌套的
# images_kwargs / videos_kwargs / audio_kwargs。vLLM 的 token 预算与
# dummy-input 读取只认扁平键，因此需要按模态叠加。
_HF_MODALITY_PROCESSOR_KWARGS = {
    "image": "images_kwargs",
    "video": "videos_kwargs",
    "audio": "audio_kwargs",
}
```

```
def overlay_modality_mm_kwargs(
    kwargs: Mapping[str, object],
    modality: str | None,
) -> dict[str, Any]:
    """将 HF 风格嵌套 processor kwargs 叠加到扁平命名空间。

    只叠加请求的模态，保证视频 size 上调不会污染图像预算；
    modality 为 None 时保持原共享命名空间行为。
    """
    merged = dict(kwargs)
    if modality is None:
        return merged

    scoped_key = _HF_MODALITY_PROCESSOR_KWARGS.get(modality)
    if scoped_key is None:
        return merged

    scoped = merged.get(scoped_key)
    if not isinstance(scoped, Mapping):
        # 非 dict 的 scoped 值直接忽略，避免字符串等被误当配置展开
        return merged

    # 叠加时保留原嵌套键，HF 侧仍能读到 videos_kwargs 等结构
    return merged | dict(scoped)
```

```
# InputProcessingContext 上的合并入口
def get_merged_mm_kwargs(
    self,
    kwargs: Mapping[str, object],
    *,
```

```

    modality: str | None = None,
) -> dict[str, Any]:
    """合并配置与请求维度的 mm_processor_kwargs。

    modality 非 None 时，把嵌套的 images_kwargs / videos_kwargs /
    audio_kwargs 叠加到扁平命名空间，供 vLLM 侧读取（token 预算、
    dummy 输入）；processor 构造与 HF __call__ 应省略 modality，
    让嵌套 dict 原样传给 HF processor。
    """

    mm_config = self.model_config.get_multimodal_config()
    merged = mm_config.merge_mm_processor_kwargs(kwargs)
    return overlay_modality_mm_kwargs(merged, modality)

```

tests/models/multimodal/processing/test_qwen3_vl.py

端到端回归：用 issue 的 CPU 探测脚本验证 videos_kwargs 只抬视频预算、flat 仍双模态共享，防止 #52834 复现。

```

# Qwen3-VL / Qwen3.8 长视频理解的像素预算来自模型卡，
# 用于复现 #52834 中 --mm-processor-kwags 的作用域问题
_LONG_VIDEO_SIZE = {"longest_edge": 469762048, "shortest_edge": 4096}

```

```

def _probe_mm_token_budgets(
    model_id: str, mm_processor_kwargs: dict[str, Any] | None
) -> tuple[int, int]:
    ctx = build_model_context(
        model_id,
        mm_processor_kwargs=mm_processor_kwargs,
        limit_mm_per_prompt={"image": 1, "video": 1},
    )
    info = MULTIMODAL_REGISTRY.create_processor(ctx.model_config).info
    video = info.get_max_video_tokens(
        seq_len=500000, mm_counts={"video": 1, "image": 1}
    )
    return video, info.get_max_image_tokens()

```

```

@pytest.mark.skip_global_cleanup
@pytest.mark.parametrize("model_id", [MODEL_ID])
def test_processor_kwargs_videos_kwargs_does_not_leak_into_image_budget(
    model_id: str,
) -> None:
    """videos_kwargs.size 必须只抬高视频 token 预算。

```

扁平 size 仍同时作用于两种模态（保持旧共享命名空间行为），
作为 #52834 的回归测试。

```

"""
stock_video, stock_image = _probe_mm_token_budgets(model_id, None)
scoped_video, scoped_image = _probe_mm_token_budgets(

```

```
    model_id, {"videos_kwargs": {"size": _LONG_VIDEO_SIZE}}
)
flat_video, flat_image = _probe_mm_token_budgets(
    model_id, {"size": _LONG_VIDEO_SIZE}
)

assert scoped_video == flat_video
assert scoped_video > stock_video
assert scoped_image == stock_image
assert flat_image > stock_image
```

评论区精华

Review 评论为 0 条，维护者 DarkLight1337 直接批准（“Thanks for fixing!”），说明方案对上了 issue 描述的两条失败路径。Issue 评论区的主要讨论是 CI 分类与 pre-commit 修复：作者将 Buildkite #85589 的两个失败任务归为无关（H200 上 Kimi K3 的 CUDA illegal-address 是已知问题、正被 #53774 回退；MI300 上内存日志断言抖动来自另一进程释放显存），维护者据此 `/ci retry` 只重跑失败步骤；mergify 两次提示 pre-commit 失败，作者补跑 `pre-commit run --all-files` 后修复。此外 commit 14891514 记录了一个接口类型权衡：`get_merged_mm_kwargs` 若声明返回 `dict[str, object]`，会让 GLM4 / Eagle 调用点的 `int()` 与 `**kwargs` 被 mypy 拒绝，因此最终放宽为 `dict[str, Any]`。

- CI 失败任务与本 PR 无关的判定 (other): 维护者 DarkLight1337 执行 `/ci retry` 只重跑失败步骤。
- `get_merged_mm_kwargs` 返回类型收紧引发 mypy 回归 (design): 改为 `dict[str, Any]` 保持旧调用点兼容，代价是静态检查强度下降。
- pre-commit 检查失败与修复 (style): 作者补跑 pre-commit 并提交修复。

风险与影响

- 风险：
 - 语义优先级新增：当扁平 `size` 与 `videos_kwargs.size` 同时存在时，`video` 读取以 `scoped` 为准；这是新增的优先级规则，已有单测覆盖，但用户可能对“flat 仍共享、scoped 优先”并存感到意外，建议在文档中明确。
 - 覆盖范围有限：目前只有 Qwen2/3-VL 的读取点接入。GLM4 / Eagle 等模型若使用同样嵌套 `kwargs`，vLLM 自身读取仍只看扁平键，`scoped` 配置仍会被吞；该模式可复用，但需要后续逐个模型接入。
 - 类型放宽：`dict[str, Any]` 降低了静态检查强度，是兼容旧调用点的代价。
 - 嵌套键残留：`overlay` 后的 `dict` 仍包含 `videos_kwargs` 等键；当前读取点只按名取 `size / min_pixels / max_pixels` 不会冲突，但若未来有人把整个 `dict` 以 `**kwargs` 展开传给下游，需要留意去重。
 - 回归面：改动集中在多模态处理上下文与两个 VL 模型，生成、采样、checkpoint 加载路径未触碰；CI 两次运行的相关 Qwen / multimodal 任务均通过，失败项均为无关已知问题。

- 影响：
 - 用户侧：Qwen3-VL / Qwen3.8 长视频 recipe 现在可以直接用 `--mm-processor-kwargs '{"videos_kwargs": {"size": ...}}'` 生效，无需手工改 checkpoint 里的 `video_preprocessor_config.json`；消除因编码器缓存低估导致的饥饿 / 活锁，以及 448K 图像 dummy 的分钟级 profiling 与启动中止。
 - 系统侧：encoder 缓存预算与运行时 token 数对齐，对 Qwen2/3-VL 的混合 image + video 推理更稳健，预留在开机 profiling 阶段就与 HF 实际处理一致。
 - 团队侧：沉淀了“HF 嵌套语义 ↔ vLLM 扁平读取”的通用对齐模式，`overlay_modality_mm_kwargs` 可被后续其他多模态模型直接复用，减少同类 bug 的重复修复成本。
 - 影响程度：中等偏低；改动面仅限多模态处理核心与两个 VL 模型，未触碰生成与采样路径，向后兼容（flat 行为不变）。
 - 风险标记：配置语义优先级变化，仅 Qwen2/3-VL 接入，返回类型放宽为 Any，嵌套键残留需留意

关联脉络

- PR #52754（标题未在材料中提供，issue 正文引用）：Issue #52834 引用了该 PR 的 review 指导，说明 `--mm-processor-kwargs` 应对应 HF processor 接受的参数形态；本 PR 是在该契约下的补齐。
- PR #51603（标题未在材料中提供，issue 正文引用）：Issue #52834 引用为 encoder cache starvation / livelock 已知故障类，本 PR 修复其配置侧根因。
- PR #52835（标题未在材料中提供，issue 正文引用）：Issue #52834 提到默认 processor cache 下 448K 图像 dummy 会中止启动的关联问题。
- PR #53774（标题未在材料中提供，CI 分类中提及）：作者在 CI 分类中说明 Kimi K3 已知问题正在该 PR 回退，与本 PR 无关。