

# PR #51408 完整报告

vllm-project/vllm

[1/N] Harden Transformers modelling backend multi-modal path

合并时间: 2026-08-08 01:30

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/51408>

## 执行摘要

- 一句话: 加固 Transformers 多模态处理, 修复占位符与字段归属 bug
- 推荐动作: 值得精读。核心设计是 `_partition_keys_by_modality`: 利用子处理器声明的 `model_input_names` 做字段归属, 既避免硬编码键名又提供未认领告警。理解该模式有助于后续处理视频模态或新增 HF 处理器兼容层。建议关注 [1/N] 系列后续 PR, 特别是 `text_replacement_offsets` 对相邻音频占位符的根本性修复。

## 功能与动机

PR body 列举了四个已修复的问题: 文本仅 prompt 会崩溃 (`KeyError: "Modality 'image' not found"`)、Token-ID prompt 重编码时特殊 token 被加两次 (Gemma3 [2] 变 [2, 2])、内存 profiling 为 3 个音频项只分配 1 个占位符 (`get_dummy_text` 无分隔符导致 run-length 扫描合并)、音频与图像字段配置互相覆盖。此外 Transformers 后端之前被排除在 `_test_processing_correctness` 之外, 没有任何 cached-vs-uncached 覆盖。

## 实现拆解

1. 修复占位符触发条件: `vllm/model_executor/models/transformers/multimodal.py` 的 `apply()` 中, `_apply_audio/_apply_vision` 的调用条件从模型是否支持该模态 (`_is_audio_model()/_is_image_model()`) 改为本次输入是否实际包含该模态项 (`mm_items.get_count("audio", strict=False)`), 从根源消除文本仅 prompt 崩溃; `_apply_vision` 的 `mm_token_type_ids is None` 守卫因 HF 返回全零张量而失效的问题也随之避免。
2. 修复 Token-ID prompt: `apply()` 在把 token IDs 解码为字符串后, 设置 `tokenization_kwargs["add_special_tokens"] = False`, 防止重新 tokenize 时再次插入特殊 token, 保证 ID 与文本 prompt 一致。
3. 修复音频占位符合并与字段覆盖: `get_dummy_text` 用空格连接多个 `audio_token`, 使 run-length 扫描能区分相邻占位符; `_apply_audio` 新增 `num_audios` 参数, 检测到的 run 数不匹配时抛出 `ValueError` (不再静默合并), 同时移除冗余的 `is_embed=True` mask; 新增 `_get_modality_field_names` 与 `_partition_keys_by_modality`, 按子处理器 `model_input_names` 把 hf\_inputs 的 key 归属到具体模态, 重写 `_get_mm_fields_config`, 避免音频与图像分支互相覆盖, 未认领 key 仅告警。
4. 测试配套: `test_common.py` 新增 `test_processing_correctness_transformers` (参数化所有 Transformers 多模态模型 ID, 覆盖三个 hit\_rate), 把该后端纳入处理一致性测试

；test\_transformers\_audio.py 重构出 \_process\_granite\_speech(separator)，新增相邻输入抛 ValueError、音频字段不被图像分支抢占、未认领字段告警测试；  
test\_transformers\_image.py 重构出 \_process\_two\_images(separator)，新增相邻图像隔离与文本仅 prompt 测试。

关键文件：

- vllm/model\_executor/models/transformers/multimodal.py（模块 多模态处理；类别 source；类型 data-contract；符号 \_get\_modality\_field\_names, \_partition\_keys\_by\_modality, \_get\_mm\_fields\_config, \_apply\_audio）：核心文件：修复 apply() 占位符触发条件、Token-ID 重复特殊 token、\_apply\_audio 占位符合并，并新增按模态分区字段的 \_partition\_keys\_by\_modality 机制。
- tests/models/multimodal/processing/test\_common.py（模块 通用测试；类别 test；类型 test-coverage；符号 get\_transformers\_backend\_model\_ids\_to\_test, test\_processing\_correctness\_transformers）：新增 test\_processing\_correctness\_transformers，首次将 Transformers 后端纳入处理一致性测试，提供 cached-vs-uncached 回归覆盖。
- tests/models/multimodal/processing/test\_transformers\_audio.py（模块 音频测试；类别 test；类型 test-coverage；符号 \_process\_granite\_speech, test\_audio\_multiple\_inputs, test\_audio\_fields\_not\_claimed\_by\_image, test\_unclaimed\_fields\_warn\_rather\_than\_raise）：重构辅助函数并新增相邻音频报错、字段归属、未认领告警用例，验证核心修复。
- tests/models/multimodal/processing/test\_transformers\_image.py（模块 图像测试；类别 test；类型 test-coverage；符号 \_process\_two\_images, test\_image\_multiple\_inputs, test\_image\_adjacent\_inputs, test\_text\_only\_prompt）：新增相邻图像隔离与文本仅 prompt 用例，确保图像路径不回归。

关键符号：\_get\_modality\_field\_names, \_partition\_keys\_by\_modality, \_get\_mm\_fields\_config, \_apply\_audio, get\_dummy\_text, apply

## 关键源码片段

### vllm/model\_executor/models/transformers/multimodal.py

核心文件：修复 apply() 占位符触发条件、Token-ID 重复特殊 token、\_apply\_audio 占位符合并，并新增按模态分区字段的 \_partition\_keys\_by\_modality 机制。

```
def _get_modality_field_names(self, modality: str) -> set[str]:
    """返回指定模态的子处理器会产生的字段名集合。"""
    # TODO: 等 huggingface/transformers#44394 落地后只用 else 分支
    if modality == "audio":
        sub_processor = self.info._get_audio_processor()
    else:
        processor = self.info.get_hf_processor()
        sub_processor = getattr(processor, f"{modality}_processor", None)

    # 预计算 embedding 会绕过子处理器，因此始终视为该模态的字段
    names = {f"{modality}_embeds"}
    for name in getattr(sub_processor, "model_input_names", None) or ():
```

```

        # 配套 mask 会被输出但不一定声明在 model_input_names 中
        names.update((name, f"{name}_mask"))
    return names

def _partition_keys_by_modality(
    self,
    keys: list[str],
    modalities: list[str],
) -> dict[str, list[str]]:
    """把 HF 处理器输出的每个 key 归属到产生它的模态。"""
    if len(modalities) == 1:
        return {modalities[0]: keys}

    claimed = {m: self._get_modality_field_names(m) for m in modalities}

    owned: dict[str, list[str]] = {modality: [] for modality in modalities}
    unclaimed = []
    for key in keys:
        for modality in modalities:
            if key in claimed[modality]:
                owned[modality].append(key)
                break
        else:
            # 没有任何模态声明的 key 直接丢弃，避免后续报错
            unclaimed.append(key)

    if unclaimed:
        logger.warning_once(
            "Unable to attribute %s to any of the modalities %s, so they "
            "will not be passed to the model. Add them to the relevant "
            "sub-processor's `model_input_names` to fix this.",
            tuple(unclaimed),
            tuple(modalities),
        )

    return owned

def _get_mm_fields_config(
    self,
    hf_inputs: "BatchFeature",
    hf_processor_mm_kwargs: Mapping[str, object],
) -> Mapping[str, MultiModalFieldConfig]:
    # HF 处理器总会返回 attention_mask，但 vLLM 不需要它
    hf_inputs.pop("attention_mask", None)

    # 大小信息由 _apply_audio/_apply_vision 写入；模态没有输入时为 None
    sizes = {
        "audio": hf_inputs.get("num_audio_tokens"),
        "image": hf_inputs.get("num_image_patches"),
    }

```

```

}
modalities = [m for m, size in sizes.items() if size is not None]

size_keys = {"num_audio_tokens", "num_image_patches"}
keys = [key for key in hf_inputs if key not in size_keys]
owned = self._partition_keys_by_modality(keys, modalities)

mm_fields: dict[str, MultiModalFieldConfig] = {
    key: MultiModalFieldConfig.flat_from_sizes(modality, sizes[modality])
    for modality in modalities
    for key in owned[modality]
}

# 这些字段总是 batched, 因为 batch size 是第一维
if "audio" in modalities:
    mm_fields["num_audio_tokens"] = MultiModalFieldConfig.batched("audio")
if "image" in modalities:
    mm_fields["image_grid_thw"] = MultiModalFieldConfig.batched("image")
    # TODO: 支持 video 模态后改路由到 "video"
    mm_fields["video_grid_thw"] = MultiModalFieldConfig.batched("image")
    mm_fields["num_image_patches"] = MultiModalFieldConfig.batched(
        "image", keep_on_cpu=True
    )
return mm_fields

```

## 评论区精华

唯一的 review 讨论围绕 `get_dummy_text` 的分隔符假设：DarkLight1337 质疑默认用 `\n` 是否通用（例如单空格也可以），hmellor 回应分隔符是任意的、仅供 profiling 使用，最终改用空格。两位 reviewer（DarkLight1337、Isotr0py）均已批准，DarkLight1337 在批准时表示 "Let's get this fix in first"。

- `get_dummy_text` 中音频占位符分隔符的选择 (design): 最终采用空格分隔；该选择不影响功能，只影响 dummy profiling 的占位符识别。

## 风险与影响

- 风险：
  - 依赖 HF `model_input_names` 契约：\_partition\_keys\_by\_modality 依赖各 HF 子处理器的 model\_input\_names，HF 版本升级或处理器变更可能导致字段无法归属而被丢弃（有 logger.warning\_once 提示）。代码已标注 TODO 跟踪 [huggingface/transformers#44394](https://huggingface.co/transformers#44394)。
  - 错误模式变更：相邻音频占位符从静默合并变为显式 ValueError，对之前能跑通但结果错误的用户是破坏性变更，但更安全。
  - 测试覆盖扩大：所有 Transformers 多模态模型进入处理正确性测试，CI 时间增加；个别模型可能因网络或兼容性跳过（已有 skip 逻辑）。

- 行为回归风险：\_apply\_audio 现在要求 run 数与音频项数严格一致，若某个 HF 处理器对单个音频产生多个 run（例如内含分隔符），会误报；当前测试仅覆盖 Granite Speech，广度有限。
- 影响：用户侧：文本仅 prompt 不再崩溃，Token-ID 与文本 prompt 行为一致，多音频 / 多图像请求的占位符定位正确。系统侧：Transformers 后端多模态字段处理从 " 按模型能力猜测 " 变为 " 按实际数据归属 "，为后续新增视频模态、text\_replacement\_offsets 奠定基础。团队侧：建立了该后端的 cached-vs-uncached 一致性回归防线，类似问题未来可被通用测试捕获。
- 风险标记：核心路径变更，依赖 HF 契约，错误模式变更

## 关联脉络

- 暂无明显关联 PR