PR #49788 完整报告

vllm-project/vllm

[Model] Enable LoRA support for tower and connector in LlavaNextForConditionalGeneration

合并时间：2026-08-18 23:53

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/49788

执行摘要

- 一句话：LLaVA-NeXT 启用 tower/connector LoRA 并修复 token 计数
- 推荐动作：值得精读。两个设计决策有借鉴价值：(1) 不扩散全局接口契约，优先复用已携带 mm_kwargs 的 get_mm_lora_token_counts() 做模型特定计算；(2) 对「占位符数不可逆」的复合 token 场景，用 pixel_values 的 tile 数前向推导而非反向推断。但需注意最终测试被删，若后续继续维护 tower/connector LoRA 栈，建议补回至少一个 adapter 加载 E2E 用例。

功能与动机

关联 issue #31479 明确提出："For the remaining models we want to support adding LoRA to the tower encoder and connector, we need to implement the following 2 functions: get_num_mm_encoder_tokens / get_num_mm_connector_tokens"，并解释根因是 "the number of multi-modal tokens represented in the language model does not necessarily match the input length required by the linear layers in the vision tower or connector"。本 PR 是该长期计划中 LLaVA-NeXT 的落地。作者在 PR 描述中补充了关键证据：LLaVA-NeXT 的 anyres/unpad 会按宽高比裁剪并追加换行 token，identity 映射的占位符计数与真实塔 / 连接器 token 数可偏差数百 token，必须改为前向计算。

实现拆解

1. 接入 LoRA 接口契约

- 在 vllm/model_executor/models/llava_next.py 中，将 LlavaNextForConditionalGeneration 的基类列表扩展为 (nn.Module, SupportsLoRA, SupportsMultiModal, SupportsPP)。
- 新增类属性 packed_modules_mapping，声明语言主干侧的 qkv_proj、gate_up_proj 为合并模块，使 LoRA 栈能按 q_proj/k_proj/v_proj、gate_proj/up_proj 拆分注入，与 LlavaForConditionalGeneration 模式一致。
- 新增 get_mm_mapping()，通过 MultiModelKeys.from_string_field(...) 返回 language_model、multi_modal_projector、vision_tower 三个权重前缀，这是 LoRA 管理器定位 tower/connector 权重的入口。
- 同步调整 import：引入 SupportsLoRA、MultiModelKeys、MultiModalKwargsItem、get_vision_encoder_info。

2. 修正 token 计数的不可逆问题

- 早期版本沿用 get_num_mm_encoder_tokens / get_num_mm_connector_tokens 的恒等映射，假设占位符数等于真实 token 数，但 unpad 在 connector 之后执行，占位符数不可反推。
- review 中 linitra24 建议不要给 interfaces.py 的全局接口加 mm_data 参数，而是直接 override 已有方法 get_mm_lora_token_counts()（该方法已携带 mm_kwargs，足以做模型特定计算）。作者采纳，最终合并结果未改动 interfaces.py。

3. 实现前向 token 计数

- get_mm_lora_token_counts() 优先从 mm_kwargs["pixel_values"] 取 pixel_values.data.shape[0] 作为 tile 数。
- 通过 get_vision_encoder_info(self.config) 获取图像尺寸与单 tile token 数，再经 get_num_selected_vision_tokens(..., self.config.vision_feature_select_strategy) 折算 connector 侧选中 token 数，返回 (num_tiles * tokens_per_tile, num_tiles * selected_per_tile)。
- mm_kwargs 缺失或 pixel_values 不是 torch.Tensor 时回退为 (num_mm_embeds, num_mm_embeds)，保持旧行为兼容。

4. 文档与测试配套

- docs/models/supported_models.md 将 LlavaNextForConditionalGeneration 行的 LoRA 列从空改为 ✅。
- 测试方面：PR 描述提到在 tests/models/multimodal/processing/test_llava_next.py 补过 anyres 网格回归用例，并新增 tests/lora/test_llava_next.py；但维护者 jeejeelee 明确要求删除 LoRA 测试文件，合并结果中 changed_files_count = 2，两类测试均未落地。

关键文件：

- vllm/model_executor/models/llava_next.py（模块 模型定义；类别 source；类型 data-contract；符号 LlavaNextForConditionalGeneration, packed_modules_mapping, get_mm_mapping, get_mm_lora_token_counts）：核心实现文件：接入 SupportsLoRA、定义 packed_modules_mapping 与 get_mm_mapping，并实现基于 pixel_values 前向计算的 get_mm_lora_token_counts()，修复 unpad 场景下 token 计数不可逆问题。
- docs/models/supported_models.md（模块 文档；类别 docs；类型 documentation）：将 LlavaNextForConditionalGeneration 的 LoRA 支持列从空改为 ✅，同步用户可见能力矩阵。

关键符号：LlavaNextForConditionalGeneration.get_mm_mapping, LlavaNextForConditionalGeneration.get_mm_lora_token_counts

关键源码片段

vllm/model_executor/models/llava_next.py

核心实现文件：接入 SupportsLoRA、定义 packed_modules_mapping 与 get_mm_mapping，并实现基于 pixel_values 前向计算的 get_mm_lora_token_counts()，修复 unpad 场景下

token 计数不可逆问题。

```
# 接入 `SupportsLoRA` 后，vLLM 的 LoRA 管理器会依据 `get_mm_mapping()`
# 返回的模块前缀，把适配器挂到视觉塔、连接器与语言主干对应的线性层上。
@MULTIMODAL_REGISTRY.register_processor(
    LlavaNextMultiModalProcessor,
    info=LlavaNextProcessingInfo,
    dummy_inputs=LlavaDummyInputsBuilder,
)
class LlavaNextForConditionalGeneration(
    nn.Module, SupportsLoRA, SupportsMultiModal, SupportsPP
):
    # 语言主干侧的 packed 映射：qkv 与 gate_up 的合并列会被拆成多个
    # 独立 LoRA，与 `LlavaForConditionalGeneration` 的模式保持一致。
    packed_modules_mapping = {
        "qkv_proj": ["q_proj", "k_proj", "v_proj"],
        "gate_up_proj": ["gate_proj", "up_proj"],
    }

    def get_mm_mapping(self) -> MultiModelKeys:
        # 返回塔、连接器、语言主干三个模块的权重前缀，供 LoRA 映射定位。
        return MultiModelKeys.from_string_field(
            language_model="language_model",
            connector="multi_modal_projector",
            tower_model="vision_tower",
        )

    def get_mm_lora_token_counts(
        self,
        *,
        modality: str,
        mm_kwargs: MultiModalKwargsItem | None,
        num_mm_embeds: int,
    ) -> tuple[int, int | None]:
        # LLaVA-NeXT 的 anyres 会把图像切成多块 tile，`unpad` 在 connector
        # 之后按宽高比裁剪并追加换行 token，因此占位符数量无法反推
        # tower/connector 的真实输入长度，必须用 `pixel_values` 的 tile 数
        # 从前向推导。
        pixel_values = mm_kwargs.get("pixel_values") if mm_kwargs else None
        if pixel_values is None or not isinstance(pixel_values.data, torch.Tensor):
            # 拿不到真实 tile 数时回退为恒等映射，保持旧行为兼容。
            return num_mm_embeds, num_mm_embeds

        num_tiles = pixel_values.data.shape[0]
        # 从 HF config 读取 vision encoder 的图像尺寸与单 tile token 数，
        # 再按 `vision_feature_select_strategy` 折算 connector 侧选中 token 数。
        encoder_info = get_vision_encoder_info(self.config)
        tile_size = encoder_info.get_image_size()
        tokens_per_tile = encoder_info.get_num_image_tokens(
```

```
        image_width=tile_size, image_height=tile_size
    )
    selected_per_tile = get_num_selected_vision_tokens(
        tokens_per_tile, self.config.vision_feature_select_strategy
    )
    return num_tiles * tokens_per_tile, num_tiles * selected_per_tile
```

评论区精华

review 中最有价值的交锋集中在两点：一是 token 计数接口的归属，二是测试的策略。linitra24 建议不要改动全局接口签名：

> If you need to use mm_data when computing mm_encoder_tokens, I think you can override get_mm_lora_token_counts() instead of adding mm_data to get_num_mm_encoder_tokens() / get_num_mm_connector_tokens(). It already provides access to mm_kwargs for model-specific token count computation.

作者回复 "thanks @linitra24, have made the necessary changes!"，最终实现采用模型内 override，未触碰 interfaces.py。

测试方面 linitra24 认为：

> I think we should also have a test to verify that the LoRA adapter can actually be loaded successfully. Since this is specifically testing LoRA support, it would be better to place the test under the LoRA test directory.

但 jeejeelee 在新增的 tests/lora/test_llava_next.py 上直接要求 "please remove this test"，作者回复 "done"，测试最终被删除。

- token 计数接口应改全局签名还是模型内 override (design): 作者接受建议，改动从 interfaces.py 的全局签名调整为模型内实现 get_mm_lora_token_counts()，最终合并结果未触碰 interfaces.py。
- LoRA 测试应验证 adapter 实际加载且放入 LoRA 目录 (testing): 作者曾新增 tests/lora/test_llava_next.py，但后续被 jeejeelee 要求删除，测试未能以建议的形态落地。
- 移除 tests/lora/test_llava_next.py (testing): LoRA 测试文件被删除，PR 最终不含任何 tower/connector LoRA 回归测试。

风险与影响

- 风险：
  1. 回归保护缺失：全部测试最终被移除，后续若 get_vision_encoder_info、pixel_values.data 结构或 vision_feature_select_strategy 语义变化，tower/connector LoRA 路径的计数错误不会被自动发现。
  2. 回退路径依赖占位符语义：当 mm_kwargs 不含 pixel_values（如纯文本请求或 pixel_values 以 list 形式传入）时，get_mm_lora_token_counts 回退为 identity 映射，混合 batch 中若 tile 数不一致，可能静默产生错误的 LoRA 输入长度。

3. 实验性路径：tower/connector LoRA 在 model_manager.py 中标注为 *experimental*，会打印警告，线上使用需自行评估稳定性。
4. 依赖 get_vision_encoder_info 的契约：该函数从 HF config 推导 vision encoder 属性，llava-hf 与 Granite Vision 等变体的 config 结构不同，若返回异常 tile 数会导致 mapping 错位。 - 影响：用户侧：llava-hf/llava-v1.6-vicuna-7b-hf、llava-v1.6-mistral-7b-hf 及 Granite Vision 用户现在可通过 *enable_tower_connector_lora=True* 对视觉塔与连接器应用 LoRA，官方测例验证了端到端生成正常；未开启该选项时行为与旧版一致。系统侧：多模态 LoRA 的 lora_mapping 在 unpad 场景下首次获得精确的输入长度，避免数百 token 的偏差；同时确立了「模型内 override get_mm_lora_token_counts() 而非改全局接口」的实现范式，后续模型可复用。团队侧：为 #31479 的多模型支持计划又补齐一项，但测试删除意味着后续维护成本转移给 reviewer 与 CI 之外的隐性回归。

- 风险标记：测试未随 PR 落地，实验性 tower/connector 路径，回退路径依赖占位符语义

关联脉络

- PR #31479 [Feature]: Enable LoRA support for tower and connector in more MM models: 本 PR 是该 issue 追踪计划中 LLaVA-NeXT 的落地，issue 明确指出需实现 get_num_mm_encoder_tokens / get_num_mm_connector_tokens 等接口。
- PR #26674 初始 tower/connector LoRA 支持（Qwen VL 系列、idefics3）：issue #31479 中列出的首批实现，定义了 get_mm_mapping 与 token 计数接口模式，本 PR 沿用同一模式。
- PR #31513 LLaVA 的 tower/connector LoRA 支持：issue #31479 中列出的 LLaVA 对应实现，本 PR 将其模式推广到 LLaVA-NeXT，并修复 unpad 场景下占位符计数不可逆的缺陷。