PR #41799 完整报告

vllm-project/vllm

[MM][Gemma4] Respect max_soft_tokens in encoder budget

合并时间：2026-05-06 20:54

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/41799

执行摘要

- 一句话：修复 Gemma4 编码器预算未遵循配置的 max_soft_tokens
- 推荐动作：值得精读，尤其是 _get_max_soft_tokens 的工具化设计，展示了如何统一处理嵌套配置键以避免重复逻辑和一致性风险。测试用例也很好地覆盖了正常、边界和异常输入。

功能与动机

Gemma4 支持显式数值 max_soft_tokens（70, 140, 280, 560, 1120），运行时处理已遵循配置，但 get_mm_max_tokens_per_item() 总是返回 280。对于 560 或 1120，编码器预算低估可能导致问题。

实现拆解

1. 定义支持值与提取函数：在 gemma4_mm.py 新增模块级常量 _SUPPORTED_SOFT_TOKENS 和工具函数 _get_max_soft_tokens，从 merged_kwargs 中提取 max_soft_tokens，支持顶层和嵌套（images_kwargs）两种路径。
2. 修改预算计算：在 get_mm_max_tokens_per_item() 中调用 _get_max_soft_tokens，若返回值为有效整数则替换默认 tokens_per_image，使编码器预算与运行时一致。
3. 统一参数解析：重构 _call_hf_processor() 中已有的 max_soft_tokens 提取逻辑，替换为 _get_max_soft_tokens 以避免重复，并正确区分顶层与嵌套来源以决定是否注入 patched_mm_kwargs。
4. 新增测试覆盖：在 test_gemma4.py 中新增两个参数化测试，分别验证 get_mm_max_tokens_per_item 返回正确 token 数（包括视频 token 断言）以及嵌套 max_soft_tokens 下的 prompt 替换正确。

关键文件：

- vllm/model_executor/models/gemma4_mm.py（模块 模型执行；类别 source；类型 data-contract；符号 _get_max_soft_tokens, _SUPPORTED_SOFT_TOKENS, get_mm_max_tokens_per_item, _call_hf_processor）：核心实现文件：新增 _get_max_soft_tokens 工具函数和 _SUPPORTED_SOFT_TOKENS 常量，修改 get_mm_max_tokens_per_item 和 _call_hf_processor 以使用配置值。
- tests/models/multimodal/processing/test_gemma4.py（模块 测试；类别 test；类型 test-coverage；符号 test_get_mm_max_tokens_per_item_respects_configured_max_soft_tokens, test_get_prompt_updates_respects_nested_max_soft_tokens）：新增两个测

试函数，覆盖显式 max_soft_tokens 在预算计算和 prompt 替换中的行为，包括嵌套和异常情况。

关键符号：_get_max_soft_tokens, get_mm_max_tokens_per_item, _call_hf_processor, get_replacement_image

关键源码片段

vllm/model_executor/models/gemma4_mm.py

核心实现文件：新增 _get_max_soft_tokens 工具函数和 _SUPPORTED_SOFT_TOKENS 常量，修改 get_mm_max_tokens_per_item 和 _call_hf_processor 以使用配置值。

```
# 模块顶部新增常量和工具函数
_SUPPORTED_SOFT_TOKENS = (70, 140, 280, 560, 1120) # 显式支持的数值

def _get_max_soft_tokens(
    merged_kwargs: Mapping[str, object],
) -> tuple[object | None, bool]:
    """从合并的 mm_kwargs 中提取 max_soft_tokens，返回 (值, 是否顶层)"""
    val = merged_kwargs.get("max_soft_tokens")
    if val is not None:
        return val, True # 顶层设置

    images_kwargs = merged_kwargs.get("images_kwargs")
    if isinstance(images_kwargs, Mapping):
        return images_kwargs.get("max_soft_tokens"), False # 嵌套设置

    return None, False # 未设置

# 修改后的预算计算方法
class Gemma4ProcessingInfo(BaseProcessingInfo):
    def get_mm_max_tokens_per_item(
        self, seq_len: int, mm_counts: Mapping[str, int]
    ) -> Mapping[str, int] | None:
        config = self.get_hf_config()
        # 默认值来自 vision_config
        tokens_per_image = config.vision_config.default_output_length
        merged_kwargs = self.ctx.get_merged_mm_kwargs({})
        val, _ = _get_max_soft_tokens(merged_kwargs)
        if isinstance(val, int) and val in _SUPPORTED_SOFT_TOKENS:
            tokens_per_image = val # 使用用户配置值
        tokens: dict[str, int] = {"image": tokens_per_image}
        if config.audio_config is not None:
            processor = self.get_hf_processor()
            tokens["audio"] = processor.audio_seq_length
        # 视频预算固定
        tokens["video"] = _VIDEO_MAX_FRAMES * (_VIDEO_MAX_SOFT_TOKENS + 2 + 6)
        return tokens
```

tests/models/multimodal/processing/test_gemma4.py

新增两个测试函数，覆盖显式 max_soft_tokens 在预算计算和 prompt 替换中的行为，包括嵌套和异常情况。

```
@pytest.mark.parametrize(
    ("mm_processor_kwargs", "expected_image_tokens"),
    [
        ({}, 280), # 默认值
        ({"max_soft_tokens": 70}, 70),
        ({"max_soft_tokens": 280}, 280),
        ({"max_soft_tokens": 1120}, 1120),
        ({"images_kwargs": {"max_soft_tokens": 560}}, 560),
        ({"images_kwargs": None}, 280), # 非 dict 回退默认
        ({"images_kwargs": "not-a-dict"}, 280), # 异常值回退默认
    ],
)
@pytest.mark.parametrize("model_id", [GEMMA4_MODEL_ID])
def test_get_mm_max_tokens_per_item_respects_configured_max_soft_tokens(
    model_id: str,
    mm_processor_kwargs: dict[str, object],
    expected_image_tokens: int,
):
    """验证 get_mm_max_tokens_per_item 返回与配置匹配的图像 token 数，同时确认视频 token
    不受影响"""
    ctx = build_model_context(
        model_id,
        mm_processor_kwargs=mm_processor_kwargs,
        limit_mm_per_prompt={"image": 1, "video": 1},
    )
    processor = MULTIMODAL_REGISTRY.create_processor(ctx.model_config)
    tokens = processor.info.get_mm_max_tokens_per_item(
        seq_len=ctx.model_config.max_model_len,
        mm_counts={"image": 1, "video": 1},
    )
    assert tokens is not None
    assert tokens["image"] == expected_image_tokens
    # 视频 token 计算公式固定
    assert tokens["video"] == 32 * (70 + 2 + 6)
```

评论区精华

1. gemini-code-assist[bot]指出 images_kwargs 非 dict 时可能引发 AttributeError，且 _get_prompt_updates 未处理嵌套参数会导致推理崩溃。作者通过 _get_max_soft_tokens 的 isinstance 检查规避了类型风险，并在 _get_prompt_updates 中统一使用该函数修复了缺失。
2. Isotr0py询问视频项目是否受影响。作者确认视频预算是独立路径（固定 70 tokens/frame），该 PR 仅调整图像预算。

3. Isotr0py建议在测试中同时验证视频 token 数量以确保稳健性。作者采纳并添加了断言。

- images_kwargs 类型安全与嵌套参数一致性 (correctness): 作者通过 _get_max_soft_tokens 中的 isinstance 检查规避了类型风险，并在 _get_prompt_updates 中统一使用该函数，修复了不一致。
- 视频预算是否受 max_soft_tokens 影响 (question): 视频预算不随图像配置变化，保持独立。
- 测试中增加视频 token 断言 (testing): 测试函数新增 assert tokens["video"] == 32 * (70 + 2 + 6) 断言。

风险与影响

- 风险:
  1. 配置项类型风险：若用户传入非 dict 的 images_kwargs（如字符串），直接调用 .get() 会抛出 AttributeError。_get_max_soft_tokens 通过 isinstance 判断已规避此风险。
  2. 嵌套参数一致性：之前 _call_hf_processor 和 _get_prompt_updates 对 max_soft_tokens 的提取逻辑不一致，可能导致预算与运行时错配。统一使用 _get_max_soft_tokens 后风险降低。
  3. 回归风险：修改了 get_mm_max_tokens_per_item 的返回值计算方式，可能影响其他依赖该值的模块（如调度器内存规划）。但仅针对 Gemma4 模型，且测试已覆盖主要路径。- 影响：影响范围：仅限于使用 Gemma4 模型并配置 mm_processor_kwargs 中 max_soft_tokens 的用户。对于默认设置（280）无行为变化。对于显式设置较小值（70）可减少内存占用，设置较大值（560/1120）可避免编码器预算不足导致的崩溃。影响程度：中等。修复了潜在的推理崩溃，但仅影响特定配置场景。
- 风险标记：images_kwargs 类型校验，视频预算独立

关联脉络

- PR #40599 [MM][Gemma4] Support auto max_soft_tokens (upstream issue): PR body 明确说明 #40599 覆盖 'auto' 支持，本 PR 只处理显式数值，两者互补构成完整功能。