PR #30566 完整报告

vllm-project/vllm

Update to transformers v5

合并时间：2026-04-16 07:29

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/30566

执行摘要

- 一句话：升级核心依赖 Transformers 至 v5 版本，启用对新模型架构的支持。
- 推荐动作：强烈建议技术管理者和核心工程师精读此 PR。这不仅是依赖版本号的变动，更是一次涉及核心架构适配的系统性工程。重点关注：
  1. 关键修复的设计决策：仔细阅读 tokenizers/registry.py 中的 get_tokenizer 函数修改，理解其如何巧妙地解决 Transformers v5 带来的配置加载顺序问题，这是本次升级中最具洞察力的技术点之一。
  2. 兼容性管理策略：通过 tests/models/registry.py 学习团队如何系统性地管理大规模模型兼容性矩阵，利用版本约束和详细原因说明来优雅地降级非阻塞性失败。
  3. 变更范围感知：通过 PR 正文和修改的文件列表，全面了解哪些模块（模型加载、tokenizer、多模态处理、测试框架）受到了影响，便于后续的问题排查和功能开发。

功能与动机

该 PR 的核心动机在于“解除 Transformers v5 版本对支持最新 SOTA 模型架构的阻碍”（PR body 中“We need this upgrade as it is blocking proper support of SoTA architectures released after Transformers v5.”）。社区用户（如 gupta-abhay）也在评论中询问是否能在 v0.18.0 中使用 Transformers v5，反映出对新特性的需求。PR 作者 hmellor 在评论中提到，团队已为此次重大升级进行了长达 4 个月的兼容性准备，涉及 140 多个 PR。

实现拆解

1. 核心依赖升级与同步：
   - 入口点：更新了多个需求文件（如 requirements/test.in, requirements/nightly_torch_test.txt），将 transformers 的版本约束从 ==4.57.5 更新至 5.5.3。
   - 连带更新：为确保生态系统兼容，同步更新了 tokenizers（至 0.22.2）、peft（下限至 0.18.1）、accelerate（至 1.13.0）、mamba（至 2.3.0）、compressed-tensors（至 0.15.0）等依赖的版本，并在 CI 环境中添加了 HF_HUB_DOWNLOAD_TIMEOUT=60 环境变量以应对 huggingface-hub v1 的默认超时调整。
2. 关键模块适配性修复：
   - 模型加载器修复 (vllm/model_executor/model_loader/gguf_loader.py)：修改 find_hf_name_in_tensor_map 函数，处理 Transformers v5 中多模态模型状态字典新

增的 "model." 外层前缀，以确保权重名称与 gguf-py 预期格式匹配。

- Tokenizer 注册表修复 (vllm/tokenizers/registry.py)：在 get_tokenizer 函数中引入配置预加载逻辑（调用 vllm.transformers_utils.config.get_config），并添加了针对 HuggingFace Hub 上 tokenizer_class 配置错误的模型（如 step3_vl）的覆盖机制，直接使用 TokenizersBackend。这是为了应对 Transformers v5 在 tokenizer 初始化期间会加载模型配置的新行为。
- Gemma4 多模态适配 (vllm/model_executor/models/gemma4_mm.py)：
- 将 Gemma4AudioInputs 张量模式的维度 "s" 标记为动态 (dynamic_dims={"s"})，以支持可变长度的音频输入批次。
- 重写视频处理逻辑，将多次迭代替换 <|video|> 占位符改为一次性替换，避免因替换文本本身包含相同占位符而导致的替换错误。

3. 测试配套调整与兼容性处理：
   - 模型注册表 (tests/models/registry.py)：为大量模型（如 Plamo2ForCausalLM, SarvamMLAForCausalLM，XverseForCausalLM 等）添加了 max_transformers_version="4.57" 或类似的上限约束，并附上详细的 transformers_version_reason 说明，以在 v5 下跳过这些因上游代码不兼容而无法工作的模型测试。
   - 测试跳过与修复：在多个多模态、量化和 LoRA 相关测试文件中添加了 @pytest.mark.skip 装饰器或条件跳过逻辑，临时禁用了已知不兼容的测试（如 intern_vl, ultravox, phi4v, MiniCPMV 等），并为一些因环境变量加载顺序导致的测试失败（如 test_olmoe_lora）提供了修复。
   - 建立向后兼容性测试：添加了基于 Transformers 4.57.5 的“backward compatibility tests”，以确保主分支在升级后仍能对旧版本进行测试。
4. 基础设施与文档更新：
   - 移除了 Dockerfile 中的 --pre 安装标志，并更新了 pip-compile 生成的锁定文件。
   - 删除了一个已过时的测试函数 test_hf_transfer_auto_activation。

关键文件：

- vllm/model_executor/models/gemma4_mm.py（模块 模型层；类别 source；类型 data-contract；符号 Gemma4AudioInputs, _call_hf_processor）：修改了多模态音频输入张量模式以支持动态序列长度，并重构了视频占位符替换逻辑以避免错误，是应对新版本多模态模型处理的典型适配案例。
- vllm/tokenizers/registry.py（模块 Tokenizer；类别 source；类型 dependency-wiring；符号 get_tokenizer）：核心修复点，为了应对 Transformers v5 在加载 tokenizer 时会同时加载模型配置的新行为，增加了配置预获取和错误 tokenizer_class 的覆盖逻辑。
- vllm/model_executor/model_loader/gguf_loader.py（模块 模型加载；类别 source；类型 data-contract；符号 find_hf_name_in_tensor_map）：修复 GGUF 模型加载器以适配 Transformers v5 中多模态模型权重命名的新约定（增加外层 'model.' 前缀），确保权重正确映射。
- tests/models/registry.py（模块 模型测试；类别 test；类型 test-coverage；符号 _HfExamplesInfo）：作为测试兼容性的总控文件，大量添加了模型级别的 Transformers

版本上限约束和跳过原因，是管理本次升级造成的功能降级（临时性）的核心配置文件。

- requirements/common.txt（模块 依赖管理；类别 infra；类型 configuration）：核心依赖版本约束的最终入口点，将其中的 'transformers < 5' 约束解除，标志着项目正式接受 v5 作为标准依赖，解决了早期 review 指出的关键冲突。

关键符号：get_tokenizer, find_hf_name_in_tensor_map, _call_hf_processor

关键源码片段

vllm/model_executor/models/gemma4_mm.py

修改了多模态音频输入张量模式以支持动态序列长度，并重构了视频占位符替换逻辑以避免错误，是应对新版本多模态模型处理的典型适配案例。

```
# 文件：vllm/model_executor/models/gemma4_mm.py
# 关键变更1：音频输入张量模式适配
class Gemma4AudioInputs(TensorSchema):
    """
    音频输入张量模式。
    Transformers v5升级后，为了正确处理批处理中不同长度的音频序列，
    需要将序列维度标记为动态。
    """
    type: Literal["audio"] = "audio"
    input_features_padded: Annotated[
        torch.Tensor,
        TensorShape("bn", "s", "f", dynamic_dims={"s"}) # 将维度 "s" 标记为动态
    ]
    input_features_mask: Annotated[
        torch.Tensor,
        TensorShape("bn", "s", dynamic_dims={"s"}) # 同上
    ]

# 关键变更2：视频占位符替换逻辑重构
# 在 _call_hf_processor 方法内处理视频时，旧的迭代替换存在缺陷。
# 新的实现将每个视频的替换文本收集起来，然后一次性完成所有替换。
video_replacements: list[str] = [] # 用于收集每个视频的替换文本
for item in videos:
    # ... 处理单个视频，生成 replacement 字符串 ...
    video_replacements.append(replacement)

# 一次性替换所有 <|video|> 占位符
vt = processor.video_token
parts = prompt.split(vt, len(video_replacements)) # 分割得到 N+1 部分
# 注意: len(parts) <= len(video_replacements) + 1
parts_with_repl: list[str] = []
# 交错拼接原始部分和替换文本
for part, repl in zip(parts, video_replacements):
    parts_with_repl.extend([part, repl])
# 添加最后一段（如果有）
parts_with_repl.extend(parts[len(video_replacements):])
```

```
prompt = "".join(parts_with_repl) # 生成最终提示字符串
```

vllm/tokenizers/registry.py

核心修复点，为了应对 Transformers v5 在加载 tokenizer 时会同时加载模型配置的新行为，增加了配置预获取和错误 tokenizer_class 的覆盖逻辑。

```
# 文件：vllm/tokenizers/registry.py
# 关键变更：get_tokenizer 函数适配 Transformers v5
def get_tokenizer(
    tokenizer_name: str | Path,
    *args,
    tokenizer_cls: type[_T] = TokenizerLike,
    trust_remote_code: bool = False,
    revision: str | None = None,
    download_dir: str | None = None,
    **kwargs,
) -> _T:
    # ... 参数解析 ...

    # 1. 配置预加载：Transformers v5 在 tokenizer 初始化时会调用 AutoConfig.from_pretrained
    # 为了确保 vLLM 的自定义配置被正确注册，需要提前获取配置。
    config = None
    with contextlib.suppress(ValueError, OSError): # 对无配置路径（如LoRA适配器）静默失败
        config = get_config(
            tokenizer_name,
            trust_remote_code=trust_remote_code,
            revision=revision,
        )

    # 2. 处理 Hub 上 tokenizer_class 配置错误的模型
    model_type = getattr(config, "model_type", None) if config else None
    if model_type in _MODEL_TYPES_WITH_INCORRECT_TOKENIZER_CLASS:
        # 对于已知有问题的模型类型（如 step3_vl），直接使用通用的 TokenizersBackend
        from transformers.tokenization_utils_tokenizers import TokenizersBackend
        logger.debug("Overriding tokenizer_class to TokenizersBackend for model_type=%r",
        model_type)
        tokenizer_cls_ = TokenizersBackend
    elif tokenizer_cls == TokenizerLike:
        tokenizer_cls_ = TokenizerRegistry.load_tokenizer_cls(tokenizer_mode)
    else:
        tokenizer_cls_ = tokenizer_cls

    # 后续正常加载 tokenizer
    tokenizer = tokenizer_cls_.from_pretrained(tokenizer_name, *args, **kwargs)
    # ... 返回 tokenizer ...
```

评论区精华

- 依赖冲突风险：gemini-code-assist[bot] 和 cursor[bot] 在早期 review 中均指出，仅升级测试依赖而保持 requirements/common.txt 中 transformers < 5 的限制将导致构建冲突。决策结论：最终 PR 通过更新 common.txt 解除了该限制，使 v5 成为项目标准。
- Gemma4 视频占位符替换算法优化：DarkLight1337 对 gemma4_mm.py 中多次迭代字符串拼接的替换逻辑提出了优化建议，结论是采用了更健壮的一次性分割 - 交错替换算法，避免了替换文本自身包含占位符时引发的错误。
- Tokenizer 配置预注册的设计权衡：hmellor 对在 vllm/transformers_utils/config.py 中添加的主动注册所有自定义配置函数 _register_custom_configs() 提出了两点关键评论：

“This will always have been the case, the tokenizer was never loaded without a config. v5 is just exposing issues that were always there.” “Also, does this not defeat the point of the config registry being lazy?” 这揭示了底层设计冲突：Transformers v5 的新行为要求配置在 tokenizer 加载前就绪，这与 vLLM 原有的惰性注册模式相悖。决策结论：最终采用了一个折中方案——在 tokenizers/registry.py 的 get_tokenizer 函数中，在加载 tokenizer 前 尝试获取并预注册配置（通过 get_config()，并优雅地忽略失败），而非全局性地破坏惰性注册。这是一个重要的设计权衡。

- 依赖约束冲突与升级策略 (design): 最终 PR 更新了 common.txt，解除了 v5 的限制，确立了 v5 为项目标准版本。这反映了从“测试先行”到“全面升级”的策略转变。
- Gemma4 视频占位符替换算法优化 (correctness): 代码被修改，采纳了一次性分割 - 交错的替换算法，提升了逻辑的健壮性。
- Transformers v5 下 tokenizer 初始化与配置注册的设计权衡 (design): 采纳了折中方案：在 tokenizer 加载点（get_tokenizer）尝试获取并预注册配置，失败则静默处理。既满足了 v5 的要求，又最大程度保持了惰性注册的特性。
- 上游模型代码不兼容的临时处理 (correctness): 决定通过向测试注册表添加 max_transformers_version 和详尽的 transformers_version_reason 来临时跳过这些测试，并计划后续跟进上游修复。

风险与影响

- 风险：- 回归风险：对核心依赖进行主版本升级，可能引入未知的、未在现有测试覆盖范围内的行为变更，尤其是在模型加载、权重映射和 tokenizer 初始化等关键路径上。vllm/model_executor/model_loader/gguf_loader.py 中对权重前缀的假设性修改，如果逻辑有误，可能导致特定多模态模型加载失败。
- 功能降级（临时性）：PR 显式禁用了约 20 个模型架构的测试（如 Plamo2, OpenCUA, Xverse, Sarvam-105B 等），这意味着这些模型在 vLLM 中使用 Transformers v5 时 可能无法正常工作。虽然作者声明这是临时措施（“not a commitment to drop these architectures forever”），但在上游模型代码更新前，这些模型的支持处于中断状态。
- 兼容性风险：升级导致 huggingface-hub v1 的引入及 HF_HUB_DOWNLOAD_TIMEOUT 的调整，可能影响模型下载的稳定性，尤其是在网络条件不佳的环境中。
- 测试覆盖缺口：尽管建立了 v4 的向后兼容性测试，但大量测试在 v5 下被跳过，可能掩盖了升级引入的其他问题。

- 影响：- 对用户的影响：积极影响是 vLLM 用户现在可以使用基于 Transformers v5 构建的最新模型。负面影响是部分旧模型（如 PR 正文所列）暂时无法使用，用户需关注其依赖的模型是否在兼容列表中。
- 对系统的影响：这是 vLLM 项目的一个里程碑式变更，标志着对 HuggingFace 生态最新能力的跟进。它影响了整个模型加载、初始化和推理的底层依赖链。
- 对团队的影响：该 PR 历时长久（201 次提交）、涉及 41 个文件，并包含了大量细致的测试适配工作，体现了团队在管理重大依赖升级和保持广泛模型兼容性方面的复杂工程努力。它为未来继续修复和恢复被跳过的模型功能奠定了基础。
- 风险标记：核心依赖升级，临时功能降级，上游依赖行为变更，广泛测试跳过

关联脉络

- PR #39862 fix online fp8 for MiniCPM models: 同属模型兼容性修复。PR 30566 中跳过了 MiniCPMV 模型，而 PR 39862 则修复了另一个 MiniCPM 模型的在线 FP8 量化问题，展示了团队在升级前后对不同模型适配问题的持续处理。
- PR #39710 [Metrics] Add request_id to FinishedRequestStats to enable correlation between metrics and requests: 同属核心功能更新。PR 30566 是底层依赖的大版本升级，PR 39710 是功能增强，二者均为影响核心路径的重要变更。
- PR #38192 [Quantization][Autoround][CPU] Add W4A16 Support: 同属扩展模型 / 格式支持。PR 30566 为支持新模型解除框架限制，PR 38192 为特定硬件和量化格式添加支持，均体现了项目扩展生态系统的努力。