PR #45195 完整报告

vllm-project/vllm

[Bugfix][V1] Clean up compiled-model bytecode hooks on VllmRunner exit

合并时间：2026-06-17 11:31

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/45195

执行摘要

- 一句话：修复 V1 单进程模式下编译模型 Bytecode Hook 残留导致内存泄漏
- 推荐动作：建议阅读该 PR，特别是 weakref.finalize 和 TorchDynamo 钩子生命周期的管理方式。它展示了如何在不引入侵入式接口的前提下，实现资源的可靠释放，是 vLLM 编译模块清理机制的参考实现。

功能与动机

Issue #21073 报告在单进程模式下多次创建 LLM 会因内存不释放而 OOM。根因是注册的字节码钩子通过绑定方法持有了模型引用，导致 GC 无法回收。

实现拆解

1. 存储钩子句柄：在 TorchCompileWithNoGuardsWrapper.__init__ 中将 register_bytecode_hook 返回的 RemovableHandle 保存为 self._bytecode_hook_handle。
2. 添加 cleanup 方法：新增 cleanup() 方法，调用 handle.remove() 移除钩子。
3. 注册终结器：在 LLMEngine.__init__（非多进程分支）中通过 _get_driver_model_for_cleanup 获取模型，并利用 weakref.finalize 注册回调，在引擎回收时执行 _cleanup_instance_caches。
4. 重置 Dynamo：在 VllmRunner.__exit__ 中添加 torch._dynamo.reset()。
5. 新增测试：添加 test_llm_delete_inprocess 测试用例，验证 GPU 内存释放。

关键文件：

- vllm/v1/engine/llm_engine.py（模块 引擎层；类别 source；类型 core-logic；符号 _get_driver_model_for_cleanup, _cleanup_instance_caches）：核心引擎，新增驱模型获取和终结器注册。
- vllm/compilation/wrapper.py（模块 编译层；类别 source；类型 core-logic；符号 cleanup）：编译包装器，存储钩子句柄并新增清理方法。
- tests/v1/shutdown/test_delete.py（模块 测试；类别 test；类型 test-coverage；符号 test_llm_delete_inprocess）：新增的测试函数，验证单进程模式下内存清理。
- tests/conftest.py（模块 测试；类别 test；类型 test-coverage）：通用测试夹具 VllmRunner，退出时增加 torch._dynamo.reset()。

关键符号：cleanup, _get_driver_model_for_cleanup, _cleanup_instance_caches, test_llm_delete_inprocess

关键源码片段

vllm/v1/engine/llm_engine.py

核心引擎，新增驱模型获取和终结器注册。

```
# LLMEngine 初始化时（非多进程模式）注册终结器
# 捕获模型引用，以便在对象回收时清理字节码钩子
model = self._get_driver_model_for_cleanup()
if model is not None:
    # weakref.finalize 确保终结器在对象被 GC 时执行，
    # 避免 __del__ 可能访问已销毁模块的问题
    self._finalizer = weakref.finalize(
        self, LLMEngine._cleanup_instance_caches, model
    )

# 静态方法，遍历模型所有模块，移除 TorchCompileWithNoGuardsWrapper 的钩子
@staticmethod
def _cleanup_instance_caches(model) -> None:
    from vllm.compilation.wrapper import TorchCompileWithNoGuardsWrapper
    for module in model.modules():
        if isinstance(module, TorchCompileWithNoGuardsWrapper):
            module.cleanup()
```

vllm/compilation/wrapper.py

编译包装器，存储钩子句柄并新增清理方法。

```
# __init__ 中：存储钩子句柄以便后续移除
if envs.VLLM_USE_BYTECODE_HOOK and mode != CompilationMode.STOCK_TORCH_COMPILE:
    self._bytecode_hook_handle = (
        torch._dynamo.convert_frame.register_bytecode_hook(self.bytecode_hook)
    )
    self._compiled_bytecode: CodeType | None = None

# 新增 cleanup 方法，移除本实例注册的钩子
def cleanup(self) -> None:
    handle = getattr(self, "_bytecode_hook_handle", None)
    if handle is not None:
        handle.remove()
```

评论区精华

在原始 PR #35676 的讨论中，确认了使用 weakref.finalize 而非 __del__ 的原因：避免终结器在对象已部分销毁时访问无效属性。此外，本 PR 也声明了其局限性——不解决所有内存清理场景（如裸 del llm），专注于单进程测试场景。

- 暂无高价值评论线程

风险与影响

- 风险：风险点包括：1）weakref.finalize 可能因引用循环或不执行而不触发，但本 PR 在 VllmRunner.__exit__ 中已手动调用 torch._dynamo.reset() 作为冗余保障；2）torch._dynamo.reset() 会全局重置 Dynamo，可能影响同一进程中其他未完成的编译任务，但单进程模式下通常只有一个引擎实例；3）变更仅在非多进程分支生效，不影响默认的多进程模式。
- 影响：影响用户：使用 VLLM_ENABLE_V1_MULTIPROCESSING=0 进行单进程推理的用户，在删除 LLM 模型后 GPU 内存可正常释放，避免后续创建模型时 OOM。影响系统：新增的终结器和 dynamo.reset() 调用在正常路径下开销极低。影响团队：提供了明确的清理模式，可作为其他资源清理的参考设计。
- 风险标记：核心路径变更，弱引用终结器可能不触发，全局 torch._dynamo.reset() 影响

关联脉络

- PR #35676 [Bugfix][V1] Clean up compiled-model bytecode hooks on VllmRunner exit: 本 PR 是旧 PR #35676 的重新提交和延续，旧 PR 仍打开但经讨论决定新开 PR。
- PR #21073 [Bug]: vLLM doesn't release memory on deletion of the LLM runner with VLLM_ENABLE_V1_MULTIPROCESSING=0: 关联的 Issue，描述了需要修复的问题。