PR #51069 完整报告

vllm-project/vllm

[CI] Prune `PyTorch Compilation Unit Tests`

合并时间：2026-08-05 09:53

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/51069

执行摘要

- 一句话：精简 PyTorch 编译测试矩阵，CI 超时从 150 分钟降至 60 分钟
- 推荐动作：建议 CI/ 测试基础设施维护者精读。值得关注的设计决策：(1) 用 DynamicShapesTestCase frozen dataclass 替代 5 维参数化笛卡尔积，以 pairwise 子集保住核心组合；(2) module 级 get_eager_outputs fixture 缓存 eager 基线，避免每个编译 case 重复启动引擎；(3) 用 compilation_counter.expect 把计数器断言并入 test_save_and_load，在不损失验证强度的前提下删除两个独立测试。

功能与动机

PR body 明确指出 PyTorch Compilation Unit Tests takes way too long，当前 timeout_in_minutes: 150，并附截图展示耗时情况。目的是在不损失关键覆盖的前提下大幅削减该 CI 门禁的耗时，减少对 H200/H100 高端 GPU 资源的长期占用，加快 CI 反馈速度。

实现拆解

1. 重构动态形状编译测试矩阵（tests/compile/test_dynamic_shapes_compilation.py）：
   - get_test_models() 从 [gpt2, Qwen2-7B-Instruct, Llama-3.1-8B]（torch 2.12 额外加 Qwen3-4B）缩减为 [Qwen/Qwen3-0.6B, openai-community/gpt2]，全部改为小模型，减少每个 case 的编译与推理时间。
   - 引入 @dataclass(frozen=True) DynamicShapesTestCase（字段 model_name / shapes_type / use_aot_compile / use_bytecode_hook），__str__ 生成 模型 -shapes-aotX-hookX 形式的参数 ID，替代原先 5 个 pytest.mark.parametrize 的笛卡尔积。
   - 新增 get_dynamic_shapes_test_cases()：参考模型 Qwen3-0.6B 覆盖 BACKED / UNBACKED / BACKED_SIZE_OBLIVIOUS × aot/hook 的 6 个 pairwise 组合，其余模型只跑 BACKED + aot=False + hook=True 冒烟，组合数从 3×3×2×2×2 = 72 降为 8 个 case。
   - 抽取 generate_outputs() 与 module 级 fixture get_eager_outputs()：eager 基线按模型只启动一次 runner 并缓存输出，避免每个编译 case 都重新拉起一个 eager 引擎，这是耗时下降的主要来源之一。
   - evaluate_guards 从该 e2e 测试中移除，但保留在 test_model_specialization_with_evaluate_guards（8 组合不变），未丢失该维度覆盖；同时把 use_aot_compile 参数从字符

串 "0"/"1" 统一为布尔值。

2. 合并 AOT 缓存测试（tests/compile/test_aot_compile.py）：
   - reference_fn / reference_fn_tuple 的循环从 3000 次降到 30 次，直接削减每轮编译执行时间。
   - test_save_and_load 增强：用 compilation_counter.expect(...) 分阶段断言（save 阶段 num_aot_compiles=1 / num_aot_artifacts_saved=1 / num_aot_artifacts_loaded=0，load 阶段 0/0/1），并补充 isinstance 返回类型检查；disable_envs_cache() 调用补齐，保证环境缓存刷新语义正确。
   - 删除 test_cache_load_returns_tuple_consistency（其单 tensor 返回场景已被 test_save_and_load 的 isinstance 断言覆盖）与 test_aot_counters_on_save_and_load（计数器断言已并入 test_save_and_load），净删 101 行，避免重复启动引擎。
3. 收紧 CI 配置（.buildkite/test_areas/pytorch.yaml）：
   - PyTorch Compilation Unit Tests 步骤 timeout_in_minutes 从 150 降到 60。
   - PyTorch Compilation Unit Tests (H100) 步骤由 device: h100 + num_devices: 1 改为 device: h200_18gb（移除 num_devices），把测试迁移到 H200 队列；步骤 label 仍保留 (H100) 字样，存在命名与机型不一致的小瑕疵。

关键文件：

- tests/compile/test_dynamic_shapes_compilation.py（模块 编译测试；类别 test；类型 test-coverage；符号 DynamicShapesTestCase, str, get_dynamic_shapes_test_cases, generate_outputs）：核心测试矩阵重构：模型从 4 个大模型缩减为 Qwen3-0.6B + gpt2，引入 DynamicShapesTestCase dataclass 与 pairwise 生成器，新增 module 级 eager 基线缓存 fixture，端到端编译测试从 5 维参数化降为单 test_case 维。
- tests/compile/test_aot_compile.py（模块 AOT 编译；类别 test；类型 test-coverage；符号 test_cache_load_returns_tuple_consistency, test_aot_counters_on_save_and_load）：AOT 缓存测试合并：reference_fn 循环 3000→30，test_save_and_load 用 compilation_counter.expect 同时验证计数器与返回类型，删除 test_cache_load_returns_tuple_consistency 和 test_aot_counters_on_save_and_load。
- .buildkite/test_areas/pytorch.yaml（模块 CI 配置；类别 config；类型 configuration）：CI 配置收紧：超时从 150 分钟降到 60 分钟；(H100) 步骤迁移到 h200_18gb 机型并移除 num_devices，减少高端 GPU 占用的同时保持门禁覆盖。

关键符号：get_dynamic_shapes_test_cases, DynamicShapesTestCase, generate_outputs, get_eager_outputs, test_save_and_load, test_dynamic_shapes_compilation, test_model_specialization_with_evaluate_guards

关键源码片段

tests/compile/test_dynamic_shapes_compilation.py

核心测试矩阵重构：模型从 4 个大模型缩减为 Qwen3-0.6B + gpt2，引入 DynamicShapesTestCase dataclass 与 pairwise 生成器，新增 module 级 eager 基线缓存 fixture，端到端编译测试从 5 维参数化降为单 test_case 维。

```
# SPDX-License-Identifier: Apache-2.0
# SPDX-FileCopyrightText: Copyright contributors to the vLLM project

import tempfile
from contextlib import contextmanager
from dataclasses import dataclass

import pytest
import torch

from tests.models.utils import check_logprobs_close
from vllm import SamplingParams
from vllm.compilation.decorators import support_torch_compile
from vllm.config import CompilationConfig, VllmConfig, set_current_vllm_config
from vllm.config.compilation import CompilationMode, DynamicShapesConfig,
DynamicShapesType
from vllm.forward_context import set_forward_context
from vllm.utils.torch_utils import is_torch_equal_or_newer


def get_test_models():
    """Get list of models to test based on PyTorch version"""
    # 只保留两个小模型：Qwen3-0.6B 作为参考模型，gpt2 作为第二冒烟模型
    # 原先的 Qwen2-7B / Llama-3.1-8B 等大模型全部移除，是耗时下降的主因
    models = [
        "Qwen/Qwen3-0.6B",
        "openai-community/gpt2",
    ]
    return models


@dataclass(frozen=True)
class DynamicShapesTestCase:
    # 把原本 5 个 parametrize 维度的笛卡尔积（3 模型 × 3 shapes × 2 aot × 2 hook × 2 guards）
    # 收敛为单个 case 对象，frozen 保证不可变，可直接作为 parametrize 参数
    model_name: str
    shapes_type: DynamicShapesType
    use_aot_compile: bool
    use_bytecode_hook: bool

    def __str__(self) -> str:
        # __str__ 提供 pytest 展示友好的参数 ID：如 qwen3-0.6b-backed-aot0-hook1
        model_id = self.model_name.rsplit("/", 1)[-1]
        return (
            f"{model_id}-{self.shapes_type.value}-"
            f"aot{int(self.use_aot_compile)}-hook{int(self.use_bytecode_hook)}"
        )

def get_dynamic_shapes_test_cases():
```

```
    """Pairwise-cover compilation options and smoke-test model classes."""
    # 参考模型 Qwen3-0.6B 覆盖全部 shapes_type 与 aot/hook 的 pairwise 组合；
    # 其余模型只做 BACKED + 默认 eager/hook 冒烟，用最少 case 保住核心编译路径覆盖
    reference_model = "Qwen/Qwen3-0.6B"
    cases = [
        DynamicShapesTestCase(reference_model, DynamicShapesType.BACKED, False, True),
        DynamicShapesTestCase(reference_model, DynamicShapesType.BACKED, True, False),
        DynamicShapesTestCase(
            reference_model, DynamicShapesType.UNBACKED, False, False
        ),
        DynamicShapesTestCase(
            reference_model, DynamicShapesType.UNBACKED, True, True
        ),
        DynamicShapesTestCase(
            reference_model, DynamicShapesType.BACKED_SIZE_OBLIVIOUS, False, True
        ),
        DynamicShapesTestCase(
            reference_model, DynamicShapesType.BACKED_SIZE_OBLIVIOUS, True, False
        ),
    ]
    # 其他模型只补一个 BACKED + eager + hook 冒烟 case
    cases.extend(
        DynamicShapesTestCase(model_name, DynamicShapesType.BACKED, False, True)
        for model_name in get_test_models()
        if model_name != reference_model
    )
    return cases
```

tests/compile/test_aot_compile.py

AOT 缓存测试合并：reference_fn 循环 3000→30，test_save_and_load 用 compilation_counter.expect 同时验证计数器与返回类型，删除 test_cache_load_returns_tuple_consistency 和 test_aot_counters_on_save_and_load。

```
@pytest.mark.skipif(not is_torch_equal_or_newer("2.10.0"), reason="requires torch 2.10")
def test_save_and_load(monkeypatch: pytest.MonkeyPatch):
    # 本测试在瘦身中承担了三个职责：
    # 1. 保存后再强制从磁盘加载 AOT artifact，验证数值一致；
    # 2. 用 compilation_counter.expect 验证 save/load 两阶段的计数器命中，
    # 替代原先独立的 test_aot_counters_on_save_and_load；
    # 3. 用 isinstance 检查返回类型，替代原先独立的
    # test_cache_load_returns_tuple_consistency（单 tensor 场景）。
    with monkeypatch.context() as m:
        args = (torch.randn(10, 10),)

        with tempfile.TemporaryDirectory() as tmpdirname:
            m.setenv("VLLM_CACHE_ROOT", tmpdirname)
            m.setenv("VLLM_USE_AOT_COMPILE", "1")
            m.setenv("VLLM_USE_MEGA_AOT_ARTIFACT", "1")
            m.setenv("VLLM_USE_STANDALONE_COMPILE", "1")
```

```
disable_envs_cache()
vllm_config = make_vllm_config()
# Phase 1: 全新编译并保存，期望恰好 1 次 compile + 1 次 save
with (
    use_vllm_config(vllm_config),
    compilation_counter.expect(
        num_aot_compiles=1,
        num_aot_artifacts_saved=1,
        num_aot_artifacts_loaded=0,
    ),
):
    compiled_mod = CompiledMod(vllm_config=vllm_config)
    expected = compiled_mod(*args)
assert isinstance(expected, torch.Tensor)

disable_envs_cache()

m.setenv("VLLM_FORCE_AOT_LOAD", "1")
vllm_config = make_vllm_config()
# Phase 2: 强制走缓存加载，期望 0 次 compile、1 次 load，
# 并验证返回类型与数值都与全新编译一致
with (
    use_vllm_config(vllm_config),
    compilation_counter.expect(
        num_aot_compiles=0,
        num_aot_artifacts_saved=0,
        num_aot_artifacts_loaded=1,
    ),
):
    cached_mod = CompiledMod(vllm_config=vllm_config)
    ret = cached_mod(*args)
assert isinstance(ret, torch.Tensor)
assert cached_mod.was_aot_compile_fn_loaded_from_disk
assert torch.allclose(ret, expected)
```

评论区精华

本 PR 没有实质技术讨论。claude[bot] 提示：该 PR 来自 fork，自动化 review 被禁用，维护者可用 @claude review 触发一次性 review；随后维护者 jeejeelee 直接空评论 approve 并合并。零讨论即合并说明改动争议小、风险自证充分，也符合纯测试 /CI 类 PR 的常规节奏。

- fork 自动 review 禁用提示 (other): 无实质技术讨论；之后由维护者 jeejeelee 直接 approve。
- 维护者审批 (other): 合并。

风险与影响

- 风险：

1. 测试覆盖缩减：动态形状端到端测试移除了 evaluate_guards 维度，虽然该维度仍在 test_model_specialization_with_evaluate_guards 中保留，但真实模型 + 编译路径下不再验证 guard 行为；UNBACKED 只在小模型 Qwen3-0.6B 上测，原先由 Qwen2-7B / Llama-3.1-8B 覆盖的“大模型 + unbacked”场景丢失。删除的 test_cache_load_returns_tuple_consistency 由 test_save_and_load 的 isinstance 断言承接，tuple 输出场景仍保留独立测试，覆盖损失有限。
2. CI 超时收紧：timeout_in_minutes 从 150 降到 60，基于当前瘦身后的测试量合理；但若后续测试增长或机器负载变化，可能再次逼近超时。
3. 机型迁移：(H100) 步骤迁移到 h200_18gb 队列，label 与机型不一致；若该队列与 H100 在架构行为上不等价，可能掩盖 H100 专属问题（同为 Hopper 平台，风险较低）。
4. fixture 缓存共享：get_eager_outputs 为 module 级 fixture，跨测试缓存 eager 输出，模型输出较小时内存可控；但若测试中途失败，缓存跨测试保留，重跑时可能跳过真实 eager 计算窗口。 - 影响：对用户无影响（纯测试与 CI 配置变更）。对团队的影响：PyTorch Compilation Unit Tests 是核心 CI 门禁之一（依赖大量 vllm/ 源码目录），耗时从约 150 分钟量级明显下降，H100 队列测试迁移到 H200，减少高端 GPU 资源占用，CI 反馈更快。测试写法上提供了可复用的范式：dataclass + pairwise 参数化生成 case、module 级 eager 基线缓存 fixture、用 compilation_counter.expect 合并重复测试。 - 风险标记：测试覆盖缩减，CI 超时收紧，测试矩阵重构，机型迁移

关联脉络

- PR #51068 Prune redundant tests points in correctness_e2e/[test_sequence_parallel,test_async_tp]: 同批次 CI 测试精简（PR 编号相邻、主题一致），说明团队正在系统性地压缩 CI 测试耗时与冗余测试点。
- PR #51015 [CI] Stabilize GLM-5.2 PCP evaluation: 同为 CI 稳定性 / 效率方向调整（修复 OOM、稳定评估配置），与本次编译测试门禁的收紧相辅相成。