PR #52221 完整报告

vllm-project/vllm

[Refactor] Remove dead code for quantization

合并时间：2026-08-17 01:10

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/52221

执行摘要

- 一句话：清理量化子系统 395 行死代码，移除 W8A8 Block INT8 内核与测试
- 推荐动作：作为清理型 PR 无需精读，但值得快速浏览删除清单，了解 vLLM 量化模块淘汰路径的决策模式：先确认无调用者，再连同测试与仅服务于该路径的 import 一起删除。若你的下游代码引用了 w8a8_block_int8_matmul、Int8Params 或 is_awq_marlin_compatible，需注意此 PR 已将其移除，应改用其他量化入口或自行维护。

功能与动机

PR body 仅有一句话 "Purpose Remove dead code for quantization"，未关联 issue。结合上下文可以推断：vLLM 量化子系统迭代极快（Humming、MXFP4、compressed-tensors 等方案持续演进），旧的 W8A8 Block INT8 路径（其内核最初从 sglang 移植而来）被取代后留下大量无调用者代码，容易误导新开发者误认为相关路径仍受支持，同时增加静态分析与 lint 噪音。此次清理既是量化模块重构前的扫清障碍动作，也以删除方式显式确认了这些方案的正式退出。

实现拆解

1. 移除 W8A8 Block INT8 内核模块（核心变更）：在 vllm/model_executor/layers/quantization/utils/int8_utils.py 中整段删除 _w8a8_block_int8_matmul（Triton jit 内核）、get_w8a8_block_int8_configs（按设备名从 JSON 加载调优配置、带 functools.lru_cache）与 w8a8_block_int8_matmul（Python 封装入口），净删 223 行；同时删除仅服务于它们的 import（functools、json、os、Any、get_device_name_as_file_name），保留仍被使用的 block_dequant 与 per_token_group_quant_int8。
2. 删除配套测试：tests/kernels/quantization/test_block_int8.py 整文件删除（75 行）。该测试以 4 种 M、2 种 N、2 种 K、两种输出 dtype 做参数化组合，对照 native_w8a8_block_matmul 验证内核数值正确性；内核已删除，测试一并移除属于干净的联动。
3. 清理各量化方案中的死符号：humming.py 删除 prepare_padded_shape、may_pad_loaded_weight 及 import math；auto_awq.py 删除类方法 is_awq_marlin_compatible；kernels/linear/base.py 删除 Int8Params dataclass 与 from_layer，参数体系从三级收敛为 Params/FP8Params 两级，并同步更新 MMLinearKernel 类 docstring；inc/schemes/inc_wna16_linear.py 删除空占位类

INCXPUW4A16LinearScheme；torchao.py 删除空方法 get_scaled_act_names。

4. 清理无引用常量：fused_moe/oracle/mxfp4.py 删除 AITER_BACKENDS 元组，mxfp4_utils.py 删除 CK_MXFP4_MOE_DIM_ALIGNMENT 常量，compressed_tensors_w4a8_fp8.py 删除 W4A8_SUPPORTED_BITS；另有 compressed_tensors_w4a8_int.py、inc.py、modelopt.py、ocp_mx_utils.py 各删 1 行。
5. 验证：提交过程中共触发 3 次 Buildkite CI（#83802、#83842、#83974）均通过，sfeng33 approve；本次不新增任何代码，故无新增测试。

关键文件：

- vllm/model_executor/layers/quantization/utils/int8_utils.py（模块 量化工具；类别 source；类型 data-contract；符号 _w8a8_block_int8_matmul, get_w8a8_block_int8_configs, w8a8_block_int8_matmul, grid）：本 PR 的绝对核心：整段删除 W8A8 Block INT8 Triton 内核三件套（_w8a8_block_int8_matmul、get_w8a8_block_int8_configs、w8a8_block_int8_matmul）及 5 个仅服务于它们的 import，净删 223 行，占删除总量一半以上。
- tests/kernels/quantization/test_block_int8.py（模块 内核测试；类别 test；类型 deletion；符号 test_w8a8_block_int8_matmul）：W8A8 Block INT8 内核唯一的数值正确性测试，整文件删除（75 行）。随内核一起移除，属于合理的测试联动。
- vllm/model_executor/layers/quantization/humming.py（模块 量化框架；类别 source；类型 data-contract；符号 prepare_padded_shape, may_pad_loaded_weight）：删除 Humming 早期权重 padding 逻辑 prepare_padded_shape 与 may_pad_loaded_weight，共 22 行，并移除不再需要的 import math。
- vllm/model_executor/kernels/linear/base.py（模块 线性内核；类别 source；类型 data-contract；符号 Int8Params, from_layer）：删除 Int8Params dataclass 及其 from_layer，MMLinearKernel 参数体系从 Params/FP8Params/Int8Params 三级收敛为两级，并同步更新类 docstring。
- vllm/model_executor/layers/quantization/auto_awq.py（模块 AWQ 量化；类别 source；类型 data-contract；符号 is_awq_marlin_compatible）：删除类方法 is_awq_marlin_compatible（AWQ 配置到 Marlin 内核的兼容性判定），该方法已无调用者。
- vllm/model_executor/layers/quantization/inc/schemes/inc_wna16_linear.py（模块 INC 量化；类别 source；类型 data-contract；符号 INCXPUW4A16LinearScheme）：删除空占位类 INCXPUW4A16LinearScheme，该类仅含 pass 无任何实现。
- vllm/model_executor/layers/fused_moe/oracle/mxfp4.py（模块 MXFP4；类别 source；类型 data-contract；符号 AITER_BACKENDS）：删除 AITER_BACKENDS 元组（AITER 后端分组），该常量已无引用。
- vllm/model_executor/layers/quantization/utils/mxfp4_utils.py（模块 MXFP4；类别 source；类型 data-contract；符号 CK_MXFP4_MOE_DIM_ALIGNMENT）：删除 CK_MXFP4_MOE_DIM_ALIGNMENT 常量（CK 预编译 MXFP4 MoE GEMM 的维度对齐约束说明），已无引用。
- vllm/model_executor/layers/quantization/torchao.py（模块 TorchAO；类别 source；类型 data-contract；符号 get_scaled_act_names）：删除空方法

TorchAOConfig.get_scaled_act_names，返回空列表的默认实现已无意义。

- vllm/model_executor/layers/quantization/compressed_tensors/schemes/compressed_tensors_w4a8_fp8.py（模块 压缩张量；类别 source；类型 data-contract；符号 W4A8_SUPPORTED_BITS）：删除 W4A8_SUPPORTED_BITS 常量（由 W4A8_SUPPORTED_TYPES_MAP 派生的重复信息）。
- vllm/model_executor/layers/quantization/compressed_tensors/schemes/compressed_tensors_w4a8_int.py（模块 压缩张量；类别 source；类型 data-contract）：删除 1 行无引用内容，属于同批死代码清理。
- vllm/model_executor/layers/quantization/inc/inc.py（模块 INC 量化；类别 source；类型 data-contract）：删除 1 行冗余内容，属于同批死代码清理。
- vllm/model_executor/layers/quantization/modelopt.py（模块 模型优化；类别 source；类型 data-contract）：删除 1 行冗余内容，属于同批死代码清理。
- vllm/model_executor/layers/quantization/utils/ocp_mx_utils.py（模块 量化工具；类别 source；类型 data-contract）：删除 1 行冗余内容，属于同批死代码清理。

关键符号：block_dequant, per_token_group_quant_int8, w8a8_block_int8_matmul, _w8a8_block_int8_matmul, get_w8a8_block_int8_configs, prepare_param, prepare_moe_param, prepare_padded_shape, may_pad_loaded_weight, is_awq_marlin_compatible, Int8Params.from_layer, get_scaled_act_names

关键源码片段

vllm/model_executor/layers/quantization/utils/int8_utils.py

本 PR 的绝对核心：整段删除 W8A8 Block INT8 Triton 内核三件套（_w8a8_block_int8_matmul、get_w8a8_block_int8_configs、w8a8_block_int8_matmul）及 5 个仅服务于它们的 import，净删 223 行，占删除总量一半以上。

```
# vllm/model_executor/layers/quantization/utils/int8_utils.py（清理后保留部分）
# 本文件原包含 W8A8 Block INT8 量化内核三件套:
# - _w8a8_block_int8_matmul：Triton jit 内核，按 block 反量化后做 matmul；
# - get_w8a8_block_int8_configs：按设备名从 JSON 加载调优参数（带 lru_cache）；
# - w8a8_block_int8_matmul：Python 封装入口，负责形状校验与 grid 计算。
# 三者共 223 行，在仓库内已无任何调用点（对应量化方案已被移除），本 PR 整体删除，
# 并清理了仅服务于它们的 import（functools、json、os、Any、get_device_name_as_file_
name）。
import logging

import torch

from vllm.platforms import current_platform
from vllm.triton_utils import tl, triton

logger = logging.getLogger(__name__)


def block_dequant(
```

```
    x_q_block: torch.Tensor,
    x_s: torch.Tensor,
    block_size: list[int],
) -> torch.Tensor:
    """按 block 做反量化：把 int8 量化 tensor 乘上对应 block 的 scale。

    参数 block_size 形如 [block_n, block_k]，x_s 的 shape 为 [n_tiles, k_tiles]，
    与 x_q_block 切分出的 tile 数一一对应。
    """
    block_n, block_k = block_size[0], block_size[1]
    n, k = x_q_block.shape
    n_tiles = (n + block_n - 1) // block_n
    k_tiles = (k + block_k - 1) // block_k
    # 防御性校验：scale 的 tile 布局必须与量化 tensor 匹配
    assert n_tiles == x_s.shape[0]
    assert k_tiles == x_s.shape[1]

    # 反量化在 fp32 中进行，避免累积精度损失
    x_dq_block = x_q_block.to(torch.float32)

    # 逐 tile 乘 scale；边界 tile 用 min() 截断，避免越界
    for i in range(k_tiles):
        for j in range(n_tiles):
            x_dq_block[
                j * block_n : min((j + 1) * block_n, n),
                i * block_k : min((i + 1) * block_k, k),
            ] *= x_s[j][i]

    return x_dq_block
```

vllm/model_executor/layers/quantization/humming.py

删除 Humming 早期权重 padding 逻辑 prepare_padded_shape 与 may_pad_loaded_weight，共 22 行，并移除不再需要的 import math。

```
# vllm/model_executor/layers/quantization/humming.py（清理后保留部分）
# 本文件删除了 prepare_padded_shape（按 block 对齐计算 padding）与
# may_pad_loaded_weight（对加载权重做 torch.nn.functional.pad 补零 / 补 1）两个函数；
# 它们是 Humming 早期按 block 对齐加载权重的辅助逻辑，已被 humming_utils 与
# vllm.utils.humming 中的新实现取代，仓库内不再有调用者，故连同 import math 一并移除。
def prepare_moe_param(tensor: torch.Tensor, name: str, extra_attrs: dict[str, Any]):
    param = torch.nn.Parameter(tensor, requires_grad=False)
    # MoE 参数把 scale_type 直接映射为 quant_method 属性，供后续 kernel 选型使用
    if "scale_type" in extra_attrs:
        extra_attrs["quant_method"] = extra_attrs["scale_type"]

    # 根据输入 / 输出维度推断是否为转置布局（用于后续 kernel 选型）
    if "input_dim" in extra_attrs and "output_dim" in extra_attrs:
        input_dim = extra_attrs["input_dim"]
        output_dim = extra_attrs["output_dim"]
```

```
        extra_attrs["is_transposed"] = input_dim < output_dim

    set_weight_attrs(param, extra_attrs)
    param.param_name = name
    return param
```

vllm/model_executor/kernels/linear/base.py

删除 Int8Params dataclass 及其 from_layer，MMLinearKernel 参数体系从 Params/FP8Params/Int8Params 三级收敛为两级，并同步更新类 docstring。

```
# vllm/model_executor/kernels/linear/base.py（清理后保留部分）
# Int8Params 及其 from_layer 在本 PR 中被删除：MMLinearKernel 的通用参数体系
# 精简为 Params（基类）与 FP8Params（fp8 专用）两级，类 docstring 同步更新，不再提及
Int8Params。
@dataclass
class Params:
    """量化线性层参数基类：weight + weight_scale + 可选 input_scale"""
    weight: torch.Tensor
    weight_scale: torch.Tensor | None
    input_scale: torch.Tensor | None

    WEIGHT: ClassVar[str] = "weight"
    WEIGHT_SCALE: ClassVar[str] = "weight_scale"
    INPUT_SCALE: ClassVar[str] = "input_scale"

    # 从 torch.nn.Module 层提取参数，供 MMLinearKernel 子类复用
    @classmethod
    def from_layer(cls, layer: torch.nn.Module) -> Self:
        return cls(
            weight=getattr(layer, cls.WEIGHT),
            weight_scale=getattr(layer, cls.WEIGHT_SCALE),
            input_scale=getattr(layer, cls.INPUT_SCALE, None),
        )


@dataclass
class FP8Params(Params):
    """FP8 层参数：在基类上增加 input_scale_ub（上界 scale）"""
    input_scale_ub: torch.Tensor | None

    INPUT_SCALE_UB: ClassVar[str] = "input_scale_ub"

    @classmethod
    def from_layer(cls, layer: torch.nn.Module) -> "FP8Params":
        """从层中提取参数"""
        return cls(
            weight=getattr(layer, cls.WEIGHT),
            weight_scale=getattr(layer, cls.WEIGHT_SCALE),
            input_scale=getattr(layer, cls.INPUT_SCALE, None),
```

```
            input_scale_ub=getattr(layer, cls.INPUT_SCALE_UB, None),
        )
```

评论区精华

该 PR 几乎没有评审争议或设计讨论。

> 审核人 sfeng33 直接给出空的 APPROVED，未提出任何问题。
>
> claude[bot] 仅发送仓库配置的自动提示，声明本仓库配置为人工 review，非实质讨论。
>
> 其余 6 条评论全部是 /ci run 触发指令与对应的 Buildkite 链接，说明整个 PR 的过程集中在 CI 验证上，代码本身被认定为无争议的机械清理。

- 无实质 review 讨论，直接 approve (other): 死代码清理无争议，审核人直接批准。

风险与影响

- 风险：核心风险是 ' 误删仍被引用的符号 ' 导致 ImportError/AttributeError。w8a8_block_int8_matmul、AutoAWQConfig.is_awq_marlin_compatible、Int8Params 等属于对外可见的半公共 Python 符号，虽然仓库内已无调用者（CI 三次通过验证了这一点），但依赖 vLLM 内部量化 API 的下游扩展可能因 import 失败被破坏——vLLM 对这些内部 API 无兼容性承诺，下游需自行适配。其次，test_block_int8.py 是 W8A8 Block INT8 内核唯一的回归保护，测试随实现一并删除后，该路径无法低成本回退（回退需同时恢复实现与测试），若未来重新引入该方案需重新移植测试。最后，如果存在 getattr 或字符串派发式调用（如按字符串查找量化方法）未被 CI 覆盖，存在低概率漏检；但本次有三次 CI 与人工审核兜底，风险可控。
- 影响：对最终用户无运行时行为影响，模型推理与量化权重加载路径完全不变。对系统而言，代码库净减 395 行，量化模块的 import 依赖、类型检查噪音与 lint 面明显减少；MMLinearKernel 参数体系从三级收敛为 Params/FP8Params 两级，新开发者理解成本降低。对团队而言，这是一次 ' 方案正式下线 ' 的存档动作：W8A8 Block INT8 路径的退出被显式记录在 git 历史中，后续量化模块重构不再需要绕过这些'历史包袱'。
- 风险标记：删除半公共 API 符号，测试随实现一并删除，跨 14 文件清理

关联脉络

- PR #52326 [CI] Shard Humming H100 eval: 间接关联：同属 Humming 量化方案推进线。本 PR 删除 Humming 遗留死代码，是该方案在 vLLM 中正式接管后的收尾动作。
- PR #52327 [CI] Shard MoE refactor B200 eval: 间接关联：MoE 量化重构的 CI 配套工作，与本 PR 对旧量化路径的清理同处于量化子系统重构周期。