PR #41892 完整报告

vllm-project/vllm

[Bugfix][Quark] Fix W8A8 INT8 garbage outputs on Step-3.5-Flash (and other 3-key fused-MoE Quark exports)

合并时间：2026-05-13 19:59

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/41892

执行摘要

- 一句话：修复 Quark INT8 W8A8 在 Step-3.5-Flash 上的垃圾输出
- 推荐动作：值得精读。展示了如何通过逐层排查独立 bug（模型配置缺失、数据布局不匹配）解决跨框架量化兼容性问题；_scale_weight_loader 和 replace_parameter 的用法可作为后续量化后端适配的参考模式。

功能与动机

PR body 指出：用 AMD Quark 量化 Step-3.5-Flash 后，vLLM 加载产生垃圾 token 和 NaN logits；相同权重转成 compressed-tensors 格式则正常，说明 bug 在 Quark 加载路径。本 PR 修复了三个根本原因。

实现拆解

1. 模型类添加 packed_modules_mapping（step3p5.py）：为 Step3p5ForCausalLM 添加 packed_modules_mapping = {"qkv_proj": [...], "gate_up_proj": [...]}，使 should_ignore_layer 能正确匹配融合模块前缀，避免共享专家被误量化。
2. MoE 路径兼容 Quark 的 per-channel scale 布局（quark_moe.py）：在 process_weights_after_loading 中检测 2D 权重 scale [E, N] 并 unsqueeze 为 [E, N, 1] 以匹配 int8 MoE 内核期望；get_fused_moe_quant_config 中为 per-channel + dynamic 场景返回 int8_w8a8_moe_quant_config，并传递 per_act_token_quant=True。同时保留旧版 per-tensor 路径。
3. 非 MoE 线性层对齐 scale 形状（quark_w8a8_int8.py）：新增 _scale_weight_loader 闭包，在加载时将 1D [N] 的 per-channel scale unsqueeze 为 [N, 1]；参数分配时也改为 [N, 1] 形状。同时将 weight_zero_point 的形状和加载器做相同处理，保证与 compressed-tensors 布局一致。
4. 配套改动：在 quark_moe.py 导入 int8_w8a8_moe_quant_config；无测试文件变更。

关键文件：

- vllm/model_executor/layers/quantization/quark/schemes/quark_w8a8_int8.py（模块 量化层；类别 source；类型 data-contract；符号 _scale_weight_loader）：添加 _scale_weight_loader 闭包，将加载时的 1D per-channel scale 变为 [N,1]；同时修改 weight_scale 和 weight_zero_point 的形状为 2D，并统一使用 _scale_weight_loader 加

载器，实现与 compressed-tensors 布局的对齐。

- vllm/model_executor/layers/quantization/quark/quark_moe.py（模块 量化层；类别 source；类型 data-contract；符号 process_weights_after_loading, get_fused_moe_quant_config）：重写 QuarkW8A8Int8MoEMethod 以支持 per-channel + dynamic per-token 量化：在 process_weights_after_loading 中将 scale 从 [E,N] 转为 [E,N,1]；在 get_fused_moe_quant_config 中为 per-channel 场景调用 int8_w8a8_moe_quant_config。
- vllm/model_executor/models/step3p5.py（模块 模型定义；类别 source；类型 data-contract；符号 Step3p5ForCausalLM.packed_modules_mapping）：为 Step3p5ForCausalLM 添加 packed_modules_mapping，使量化配置中的 exclude 列表能正确匹配融合后的模块名，避免共享专家层被误认为需要量化。

关键符号：_scale_weight_loader, process_weights_after_loading, get_fused_moe_quant_config

关键源码片段

vllm/model_executor/layers/quantization/quark/schemes/quark_w8a8_int8.py

添加 _scale_weight_loader 闭包，将加载时的 1D per-channel scale 变为 [N,1]；同时修改 weight_scale 和 weight_zero_point 的形状为 2D，并统一使用 _scale_weight_loader 加载器，实现与 compressed-tensors 布局的对齐。

```
# vllm/model_executor/layers/quantization/quark/schemes/quark_w8a8_int8.py

def create_weights(self, layer, output_partition_sizes, input_size_per_partition,
                   params_dtype, weight_loader, **kwargs):
    layer.logical_widths = output_partition_sizes

    # Quark 将 per-channel weight_scale 存为 1D [N]；此处 reshape 为 [N, 1]
    def _scale_weight_loader(param, loaded_weight, *args, **kwargs):
        if loaded_weight.dim() == 1:
            loaded_weight = loaded_weight.unsqueeze(-1)
        return weight_loader(param, loaded_weight, *args, **kwargs)

    self.kernel = init_int8_linear_kernel(...)
    # ... weight parameter ...
    if self.qscheme == "per_channel":
        weight_scale = ChannelQuantScaleParameter(
            data=torch.empty((sum(output_partition_sizes), 1), dtype=torch.float32),
            output_dim=0,
            weight_loader=_scale_weight_loader, # 使用新的加载器
        )
        weight_zero_point = ChannelQuantScaleParameter(
            data=torch.empty((sum(output_partition_sizes), 1), dtype=torch.int8),
            output_dim=0,
            weight_loader=_scale_weight_loader,
```

```
        )
    else: # per_tensor 保持不变
        weight_scale = PerTensorScaleParameter(...)
        weight_zero_point = PerTensorScaleParameter(...)
    layer.register_parameter("weight_scale", weight_scale)
    layer.register_parameter("weight_zero_point", weight_zero_point)
```

vllm/model_executor/layers/quantization/quark/quark_moe.py

重写 QuarkW8A8Int8MoEMethod 以支持 per-channel + dynamic per-token 量化：在 process_weights_after_loading 中将 scale 从 [E,N] 转为 [E,N,1]；在 get_fused_moe_quant_config 中为 per-channel 场景调用 int8_w8a8_moe_quant_config。

```
# vllm/model_executor/layers/quantization/quark/quark_moe.py

# 在 process_weights_after_loading 中，对 per-channel 权重 scale 进行 reshape
if self.weight_qscheme == "per_channel":
    for attr in ("w13_weight_scale", "w2_weight_scale"):
        param = getattr(layer, attr, None)
        if param is not None and param.dim() == 2:
            replace_parameter(
                layer,
                attr,
                torch.nn.Parameter(
                    param.data.unsqueeze(-1).contiguous(), # [E, N] -> [E, N, 1]
                    requires_grad=False,
                ),
            )

# 在 get_fused_moe_quant_config 中，对 per-channel + dynamic 场景做特殊处理
if self.weight_qscheme == "per_channel" and not self.static_input_scales:
    return int8_w8a8_moe_quant_config(
        w1_scale=layer.w13_weight_scale,
        w2_scale=layer.w2_weight_scale,
        a1_scale=layer.w13_input_scale,
        a2_scale=layer.w2_input_scale,
        w1_bias=getattr(layer, "w13_bias", None),
        w2_bias=getattr(layer, "w2_bias", None),
        per_act_token_quant=True,
    )
# 后续保留原有 per-tensor / static 逻辑作为 fallback
```

vllm/model_executor/models/step3p5.py

为 Step3p5ForCausalLM 添加 packed_modules_mapping，使量化配置中的 exclude 列表能正确匹配融合后的模块名，避免共享专家层被误认为需要量化。

```
# vllm/model_executor/models/step3p5.py

class Step3p5ForCausalLM(nn.Module, SupportsPP, MixtureOfExperts):
    # 必须定义 packed_modules_mapping，否则 Quark/CT 的 exclude 列表无法
```

```
# 反向映射到 fused 模块名（如 gate_up_proj），导致共享专家层被误量化。
packed_modules_mapping = {
    "qkv_proj": ["q_proj", "k_proj", "v_proj"],
    "gate_up_proj": ["gate_proj", "up_proj"],
}
# ... 其余定义
```

评论区精华

1. replace_parameter 建议（gemini-code-assist[bot]）：在 quark_moe.py 中建议使用 replace_parameter 替代 setattr 以保留自定义属性。作者采纳，最终代码使用了 replace_parameter。
2. 精简注释（tjtanaa）：指出 quark_w8a8_int8.py 中注释过多，建议删除批量注释。作者响应后简化了注释。

- 使用 replace_parameter 替代 setattr (design): 作者采纳建议，最终代码使用了 replace_parameter。
- 精简注释 (style): 作者响应并简化了注释，保留必要说明。
- 请求提供模型链接 (question): 已提供模型链接，便于验证。

风险与影响

- 风险：风险低：变更严格限定在 Quark INT8 量化路径和 Step3p5ForCausalLM 模型；compressed-tensors 路径未改动。Per-tensor 静态量化的旧路径完全保留，不会退化。但缺少直接针对此 fix 的测试用例，回归依赖现有测试套件。
- 影响：对使用 AMD Quark 导出 INT8 检查点的用户影响显著：直接提供关键修复，使 Step-3.5-Flash 等 MoE 模型的 INT8 量化推理可用；对其他模型（如非 fused-MoE 架构）无影响。团队内部需关注 Quark 路径的测试覆盖。
- 风险标记：缺少测试覆盖，量化路径特异性修复

关联脉络

- PR #36320 previous Quark INT8 fix (mentioned in PR body): PR body 提到 follow-up to #36320，但未在历史 PR 列表中；可推断为关联的前置 PR。