PR #2256 完整报告

THUDM/slime

fix(tools): clamp block max in block_fp8 to avoid NaN weights from all-zero blocks

合并时间：2026-08-12 13:28

原文链接：http://prhub.com.cn/THUDM/slime/pull/2256

执行摘要

- 一句话：修复 block_fp8 全零块产生 NaN 权重的问题
- 推荐动作：该 PR 值得精读，尤其是对于使用 block_fp8 进行模型量化的团队。它展示了一个典型边界条件的修复，并通过引入针对性的单元测试强化了代码契约。建议关注测试的编写方式（动态加载模块、CPU 运行）以及钳制值选择的一致性。

功能与动机

block_fp8 是转换脚本的默认量化策略，但其计算 scale 时未将 block max 下限钳制到非零值，这与同文件中的 channel_fp8 和 tensor_fp8（均使用 clamp(min=1e-12)）不一致。对于大型 MoE 检查点中常见的全零 128×128 块（如 padding vocab 行、未使用的 expert、被置零的 gate 权重），block_max == 0 会导致 scale == 0，进而 qweight == 0 / 0 == NaN，NaN 被静默写入 safetensors，反量化时污染模型前向传播。该 PR 旨在通过钳制 block max 修复此问题，确保转换后的模型不会因 NaN 权重而受损。

实现拆解

1. 修改 tools/convert_hf_to_fp8.py 中 block_fp8 函数的 scale 计算，将 block_max.to(torch.float32) / FP8_MAX 改为 block_max.clamp(min=1e-12).to(torch.float32) / FP8_MAX，与 channel_fp8、tensor_fp8 保持一致。
2. 新增测试文件 tests/test_block_fp8_zero_block.py，通过 importlib 动态加载转换器模块，提供三个 CPU 单元测试：
   - test_block_fp8_all_zero_block_has_no_nan：验证混合零 / 非零块输出无 NaN/inf 且所有 scale > 0。
   - test_block_fp8_zero_block_roundtrips_to_zero：验证全零块往返量化后仍为精确零。
   - test_block_fp8_nonzero_blocks_unaffected：验证非零块的量化 / 反量化误差在 FP8 容差内，防止行为回归。
3. 测试文件使用 pytest.importorskip 处理可选的 safetensors 和 torch 依赖，并设置 NUM_GPUS = 0 以便在 CPU 上运行。

关键文件：

- tools/convert_hf_to_fp8.py（模块 转换工具；类别 source；类型 core-logic；符号 block_fp8）：核心修复文件，修改 block_fp8 的 scale 计算，添加 clamp(min=1e-12) 以防止全零块产生 NaN。

- tests/test_block_fp8_zero_block.py（模块 测试；类别 test；类型 test-coverage；符号 _load_converter, converter, test_block_fp8_all_zero_block_has_no_nan, test_block_fp8_zero_block_roundtrips_to_zero）：新增测试文件，覆盖全零块、混合块和正常块的量化行为，防止回归。

关键符号：block_fp8

关键源码片段

tools/convert_hf_to_fp8.py

核心修复文件，修改 block_fp8 的 scale 计算，添加 clamp(min=1e-12) 以防止全零块产生 NaN。

```
# tools/convert_hf_to_fp8.py 中的 block_fp8 函数（修复后）
def block_fp8(weight, block_size):
    # per block quant
    block_n, block_k = block_size[0], block_size[1]
    shape_0, shape_1 = weight.shape
    n_tiles = ceildiv(shape_0, block_n)
    k_tiles = ceildiv(shape_1, block_k)

    # 对 weight 进行 padding，使尺寸对齐 block 边界，padding 部分填充 0
    q_weight = F.pad(weight, (0, k_tiles * block_k - shape_1, 0, n_tiles * block_n - shape_0),
    mode="constant", value=0.0)

    qweight = q_weight.reshape(n_tiles, block_n, k_tiles, block_k)

    # 计算每个 block 的绝对值最大值（即 block max）
    block_max = torch.max(torch.abs(qweight), dim=1, keepdim=True)[0]
    block_max = torch.max(block_max, dim=3, keepdim=True)[0]

    # 关键修复：对 block_max 进行下限钳制，避免全零块导致 scale == 0 继而产生 NaN
    # 与 channel_fp8 / tensor_fp8 保持一致
    scale = block_max.clamp(min=1e-12).to(torch.float32) / FP8_MAX

    # 量化并截断到 FP8 范围
    qweight = ( (qweight / scale).clamp(min=FP8_MIN, max=FP8_MAX)
        .reshape((n_tiles * block_n, k_tiles * block_k))
        .to(torch.float8_e4m3fn) )

    # 去掉 padding 部分，返回原始形状的量化权重和 scale
    qweight = qweight[:shape_0, :shape_1].clone().detach()
    scale = scale.reshape(n_tiles, k_tiles)
    return qweight, scale
```

tests/test_block_fp8_zero_block.py

新增测试文件，覆盖全零块、混合块和正常块的量化行为，防止回归。

```
# tests/test_block_fp8_zero_block.py - 关键测试片段
```

```
def test_block_fp8_all_zero_block_has_no_nan(converter):
    # 构造一个 256x256 的 bf16 权重，只有第一个 128x128 tile 非零，其余为全零块
    weight = torch.zeros(256, 256, dtype=torch.bfloat16)
    weight[0, 0] = 1.0

    qweight, scale = converter.block_fp8(weight, (128, 128))

    # 断言量化结果不含 NaN 或 inf
    assert not torch.isnan(qweight.float()).any()
    assert not torch.isinf(qweight.float()).any()

    # 断言所有 scale 均大于 0，避免反量化时乘以 0 产生 NaN
    assert (scale > 0).all()

def test_block_fp8_zero_block_roundtrips_to_zero(converter):
    # 全零块应往返还原为精确零
    weight = torch.zeros(128, 128, dtype=torch.bfloat16)
    qweight, scale = converter.block_fp8(weight, (128, 128))
    dequantized = qweight.float() * scale.float()
    assert (dequantized == 0).all()
```

评论区精华

该 PR 没有评论或 review 讨论。但实现中清晰体现了与 channel_fp8 和 tensor_fp8 保持一致的设计决策，因此讨论重点（如果有）会是如何权衡钳制值的选择（1e-12 vs 其他）以及为什么默认策略此前遗漏了该保护。

- 暂无高价值评论线程

风险与影响

- 风险：该修复仅修改 block_fp8 的 scale 计算，只影响全零或近零块的量化，对非零块的影响为零。风险较低：
  - 钳制值 1e-12 足够小，不会对非零 block 的 scale 产生可感知影响。
  - 新测试覆盖了关键场景，且现有测试（如果有）未受影响。
  - 需要注意，convert_hf_to_fp8.py 是转换工具，修复可能改变转换后的权重值（仅全零块从 NaN 变为 0），但这是预期的正确行为。
- 影响：影响范围有限，主要涉及量化转换工具的使用者：
  - 修复可能影响使用 block_fp8 转换 MoE 模型（含全零块）的工具链，避免生成的模型前向传播被 NaN 污染。
  - 对已有非全零块的模型转换结果无影响。
  - 新增测试可保护该行为，防止未来回归。
  - 团队需要在后续转化流程中验证转换后的模型可用性。
  - 风险标记：边界条件修复，新增测试覆盖

关联脉络

- 暂无明显关联 PR