PR #36097 完整报告

sgl-project/sglang

Fix MXFP8 MoE weight sizing for non-gated models

合并时间：2026-08-25 22:09

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36097

执行摘要

- 一句话：修复非 gated MoE 的 MXFP8 权重尺寸分配
- 推荐动作：值得精读。该 PR 展示了如何从层配置推导张量形状而非硬编码，是量化模块中一种可复用的模式；同时覆盖了模型结构差异导致的隐蔽 bug，具有教学意义。

功能与动机

PR body 明确指出 *create_fp8_moe_weight_* 硬编码 is_concat=True，导致非 gated MoE（NemotronH 使用 relu2，checkpoint 只有 up_proj/down_proj 无 gate_proj）的 w13 总是按 2 * intermediate 分配，上半部分成为永远不会被加载器写入的 torch.empty，静默破坏量化权重。基准显示修复前 gsm8k 5-shot 准确率仅 0.066，修复后 0.9447 恢复至 bf16 参考水平。

实现拆解

1. 在 python/sglang/srt/layers/quantization/fp8.py 的 *create_fp8_moe_weight_* 中引入 w13_num_shards = 2 if layer.moe_runner_config.is_gated *else* 1，取代所有硬编码 2 的维度。
2. 将 get_moe_weight_sizes 的 is_concat 参数由固定 True 改为 layer.moe_runner_config.is_gated，使 w13_up_dim 正确反映非 gated 层的单分片布局。
3. 同步调整 w13 权重、scale、bias 的分配逻辑，包括 fp4 expert、HIP int4、block quant、per-tensor 等分支，全部改用 w13_num_shards 计算维度；process_weights_after_loading 中的 gemm stride 与 requant 循环也一并修正。
4. 新增 test/registered/unit/layers/quantization/test_fp8_moe_weight_gating.py，用 _RecordingLayer 模拟层并断言不同 gating 下的 w13 及 scale 形状，通过 register_cpu_ci 接入 CPU CI。
5. 测试覆盖 gated 融合（2 * intermediate）、非 gated 只含 up（intermediate）、block scale 与权重的行数对齐、per-tensor scale 形状等四个场景。

关键文件：

- python/sglang/srt/layers/quantization/fp8.py（模块 量化层；类别 source；类型 core-logic；符号 *create_fp8_moe_weight_*, process_weights_after_loading）：核心源码改动：修复非 gated MoE 权重与 scale 的尺寸分配，影响所有复用 *create_fp8_moe_weight_* 的量化路径。

- test/registered/unit/layers/quantization/test_fp8_moe_weight_gating.py（模块 MoE 权重；类别 test；类型 test-coverage；符号 _RecordingLayer, _create_weights, TestFp8MoEWeightGating, test_gated_fuses_gate_and_up）：新增回归测试，覆盖 gated 与非 gated 两条路径的形状断言，防止再次出现尺寸不匹配。

关键符号：create_fp8_moe_weight_, process_weights_after_loading, test_gated_fuses_gate_and_up, test_non_gated_w13_holds_up_only, test_non_gated_block_scale_matches_weight

关键源码片段

python/sglang/srt/layers/quantization/fp8.py

核心源码改动：修复非 gated MoE 权重与 scale 的尺寸分配，影响所有复用 create_fp8_moe_weight_ 的量化路径。

```
# 计算 w13 的 shard 数量：gated 层融合 gate + up 两路，非 gated 只融合 up
w13_num_shards = 2 if layer.moe_runner_config.is_gated else 1

# 将 is_concat 改为由配置推导，保证 get_moe_weight_sizes 返回正确维度
w13_up_dim, w2_up_dim, weight_padded = get_moe_weight_sizes(
    intermediate_size_per_partition,
    is_aiter_moe=_use_aiter,
    is_concat=layer.moe_runner_config.is_gated,
    is_packed=False,
)

# 权重分配：所有 w13 相关缓冲区统一使用 w13_num_shards，
# 避免非 gated 层出现未初始化的上半部分
if is_fp4_expert:
    w13_weight = torch.nn.Parameter(
        torch.empty(
            num_experts,
            w13_num_shards * intermediate_size_per_partition,
            hidden_size // 2,
            dtype=torch.int8,
        ),
        requires_grad=False,
    )
elif _is_hip and _use_hip_int4:
    w13_weight = torch.nn.Parameter(
        torch.empty(
            num_experts,
            w13_num_shards * intermediate_size_per_partition,
            hidden_size // 8,
            dtype=params_dtype,
        ),
        requires_grad=False,
    )
else:
```

```
        w13_weight = torch.nn.Parameter(
            torch.empty(num_experts, w13_up_dim, hidden_size, dtype=params_dtype),
            requires_grad=False,
        )
```

test/registered/unit/layers/quantization/test_fp8_moe_weight_gating.py

新增回归测试，覆盖 gated 与非 gated 两条路径的形状断言，防止再次出现尺寸不匹配。

```
class TestFp8MoEWeightGating(CustomTestCase):
    def test_gated_fuses_gate_and_up(self):
        # gated 层 w13 应为 gate + up 两路，尺寸为 2 * intermediate
        params = _create_weights(is_gated=True, block_quant=True)
        self.assertEqual(params["w13_weight"].shape[1], 2 * INTERMEDIATE)

    def test_non_gated_w13_holds_up_only(self):
        # 回归：非 gated 层 w13 只含 up，尺寸必须为 intermediate
        params = _create_weights(is_gated=False, block_quant=True)
        self.assertEqual(params["w13_weight"].shape[1], INTERMEDIATE)

    def test_non_gated_block_scale_matches_weight(self):
        # scale 行数必须与权重行数按 block_n 对齐
        params = _create_weights(is_gated=False, block_quant=True)
        weight_rows = params["w13_weight"].shape[1]
        scale_rows = params["w13_weight_scale_inv"].shape[1]
        self.assertEqual(scale_rows * BLOCK_N, weight_rows)
```

评论区精华

mmangkad 在 review 中质疑测试的 checkpoint 是 bf16 还是已量化 mxfp8，并担心 w13_weight_scale 仍按 2 * intermediate 分配会影响到预量化 checkpoint。elvischenv 回应：已把 fp8.py 中所有硬编码 2 个 shard 的位置都改为由 is_gated 决定，与 modelopt_quant.py 的做法一致；目前唯一的非 gated MoE 是 Nemotron 系列，且只有 modelopt 预量化，尚无 non-gated 的 mxfp8 预量化 checkpoint，本次修复的目标是 bf16 模型 + 在线 MXFP8 量化。随后 mmangkad 批准。

- 测试 checkpoint 类型与 scale 兼容性 (question): elvischenv 回应：fp8.py 中所有硬编码 2 个 shard 的地方都已改为由 is_gated 决定；当前无非 gated 的 mxfp8 预量化 checkpoint，此修复针对在线量化路径。
- 建议用预量化 checkpoint 再验证 (question): 未在 PR 内进一步回复，但 maintainer 已批准合并；该建议可作为后续跟进项。

风险与影响

- 风险：主要在量化权重分配路径，风险点包括： 1) 对 gated 模型保持兼容，因为 is_gated=True 时行为不变； 2) 非 gated 场景目前只有 NemotronH 一个模型，若未来出现预量化的非 gated MXFP8 checkpoint，可能需要额外适配； 3) layer.moe_runner_config.is_gated 的获取依赖 MoeRunnerConfig 正确配置，测试通过 mock 验证了调用点，其他量化方法若复用此函数需保证传入的 layer 有该属性。

- 影响：修复 Nemotron-3-Ultra-550B 在 GB300 + TP8 + MXFP8 量化下的严重精度回退，端到端 gsm8k 从 0.066 恢复至 0.9447。对已有 gated MoE 模型（如 DeepSeek、Qwen 等）无影响，因为默认路径不变。新增测试通过 register_cpu_ci 接入 CPU CI，防止回归，影响面可控。
- 风险标记：预量化兼容路径未验证，依赖 MoeRunnerConfig 配置，量化核心路径变更

关联脉络

- 暂无明显关联 PR