PR #31825 完整报告

sgl-project/sglang

[Quant] Support NVFP4_AWQ checkpoints in ModelOpt FP4 path

合并时间：2026-07-22 08:46

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/31825

执行摘要

- 一句话：支持 ModelOpt NVFP4_AWQ 检查点加载
- 推荐动作：该 PR 值得精读，尤其是 create_weights 和 apply 中的 AWQ 分支实现，展示了如何在现有量化路径中优雅地扩展新格式。对于关注量化推理的工程师，可学习其 per-channel scaling 的参数注册和加载策略。

功能与动机

支持使用 NVIDIA Model Optimizer 量化后的 AWQ 检查点加载和推理，降低用户服务 AWQ 量化模型的复杂度。PR body 中明确说明：「This was done to support easier serving of AWQ quantized models. I used Nvidia Model Optimizer for the quantization」。

实现拆解

1. 允许列表扩展：在 modelopt_quant.py 的 from_config 方法中，将 quant_method 允许列表从 ["FP8", "NVFP4"] 扩展为 ["FP8", "NVFP4", "NVFP4_AWQ"]，避免 NVFP4_AWQ 配置被误判为不支持。
2. is_awq 标记与参数注册：在 ModelOptFp4Config.__init__ 中新增 is_awq 参数（默认 False），在 from_config 中通过 is_awq="AWQ" in quant_method 自动设置。在 create_weights 中，当 is_awq 为 True 时，为每层注册一个 per-input-channel 的 pre_quant_scale 参数（形状 [input_size_per_partition]，input_dim=0 以便在 row-parallel 线性层中正确切分）。
3. 激活前缩放应用：在 apply 方法中，若 is_awq 为 True，在调用 fp4_quantize 之前将输入 x 逐元素乘以 layer.pre_quant_scale。非 AWQ 层不受影响。
4. group_size 默认值：在 flat config 解析路径中，如果 group_size 为 None 且 quant_method 包含 "NVFP4"（包括 NVFP4_AWQ），则默认设为 16，因为 ModelOpt 的 AWQ 导出在 flat config.json 中会省略 group_size（仅嵌套的 hf_quant_config.json 中存在）。
5. 测试补充：在 test_modelopt_loader.py 中添加 test_awq_flat_config_defaults_group_size 测试，验证 ModelOptFp4Config.from_config 能正确解析缺少 group_size 的 NVFP4_AWQ flat config，并返回 group_size=16 且 is_awq=True。同时在参数化测试 _MODELOPT_CASES 中增加 ({"quant_algo": "NVFP4_AWQ"}, "modelopt_fp4") 条目。

关键文件：

- python/sglang/srt/layers/quantization/modelopt_quant.py（模块 量化模块；类别 source；类型 data-contract）：核心变更文件：扩展 NVFP4_AWQ 支持，包括允许列表、is_awq 标记、pre_quant_scale 参数注册、前向缩放逻辑以及 flat config group_size 默认值。
- test/registered/unit/model_loader/test_modelopt_loader.py（模块 测试模块；类别 test；类型 test-coverage；符号 test_awq_flat_config_defaults_group_size）：新增单元测试，验证 NVFP4_AWQ flat config 中 group_size 默认值为 16 和 is_awq 标记正确设置。

关键符号：ModelOptFp4Config.init, ModelOptFp4Config.from_config, ModelOptFp4Config.create_weights, ModelOptFp4Config.apply

关键源码片段

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

核心变更文件：扩展 NVFP4_AWQ 支持，包括允许列表、is_awq 标记、pre_quant_scale 参数注册、前向缩放逻辑以及 flat config group_size 默认值。

```
# modelopt_quant.py: NVFP4_AWQ 支持的关键片段

class ModelOptFp4Config(ModelOptQuantConfig):
    def __init__(self, ..., group_size=None, ..., is_awq: bool = False):
        # ...
        self.is_awq = is_awq # AWQ 标记，NVFP4_AWQ 检查点设为 True
        self.group_size = group_size

    @classmethod
    def from_config(cls, config):
        # ... 解析 quant_method ...
        # 将 NVFP4_AWQ 加入允许列表
        if quant_method not in ["FP8", "NVFP4", "NVFP4_AWQ"]:
            raise ValueError("ModelOpt currently only supports: FP8, NVFP4, NVFP4_AWQ")
        # flat config 中缺失 group_size 时，NVFP4 系列默认 16
        if group_size is None and quant_method and "NVFP4" in quant_method:
            group_size = 16
        # ...
        return cls(
            ...,
            group_size=group_size,
            is_awq="AWQ" in quant_method, # 自动设置 AWQ 标记
        )

    # create_weights 中：为 AWQ 层注册 per-input-channel pre_quant_scale
    def create_weights(self, layer, ...):
        # ... 已有参数注册 ...
        if self.quant_config.is_awq:
            pre_quant_scale = ModelWeightParameter(
                data=torch.ones(input_size_per_partition, dtype=params_dtype),
                input_dim=0, output_dim=0,
```

```
                weight_loader=weight_loader,
            )
            layer.register_parameter("pre_quant_scale", pre_quant_scale)
        # ...

    # apply 中：在 fp4_quantize 之前应用 pre_quant_scale
    def apply(self, layer, x, ...):
        # ... 非 AWQ 路径不变 ...
        if self.quant_config.is_awq:
            x = x * layer.pre_quant_scale # 逐 channel 缩放激活
        x_fp4, x_scale_interleaved = fp4_quantize(x, layer.input_scale_inv)
        # ...
```

test/registered/unit/model_loader/test_modelopt_loader.py

新增单元测试，验证 NVFP4_AWQ flat config 中 group_size 默认值为 16 和 is_awq 标记正确设置。

```
# test_modelopt_loader.py: NVFP4_AWQ 解析测试

class TestParseQuantHfConfig(CustomTestCase):
    # 参数化测试新增 NVFP4_AWQ 条目
    _MODELOPT_CASES = [
        ...
        ({"quant_algo": "NVFP4_AWQ"}, "modelopt_fp4"), # 新：NVFP4_AWQ 应映射为 modelopt_
        fp4
        ...
    ]

    def test_awq_flat_config_defaults_group_size(self):
        """NVFP4_AWQ flat config.json 省略 group_size，from_config 应默认设为 16。"""
        cfg = ModelOptFp4Config.from_config({
            "quant_algo": "NVFP4_AWQ",
            "ignore": ["lm_head"],
            "quant_method": "modelopt",
        })
        self.assertEqual(cfg.group_size, 16) # 验证默认值
        self.assertTrue(cfg.is_awq) # 验证 AWQ 标记
```

评论区精华

该 PR 的 review 过程简短：BBuf 进行了批准（APPROVED），没有留下 review 评论。gemini-code-assist 自动生成了一个配额耗尽提示。讨论主要集中于通过 /tag-and-rerun-ci 触发 CI 测试。

- 暂无高价值评论线程

风险与影响

- 风险：

1. 回归风险：变更修改了 from_config 和 apply 等核心路径，可能影响现有 NVFP4（非 AWQ）模型的加载和推理。但改动范围小且逻辑清晰（仅增加分支），回归概率较低。
2. 兼容性：新增的 is_awq 参数默认 False，现有 NVFP4 检查点不受影响。pre_quant_scale 参数仅在 is_awq 为 True 时创建，不影响非 AWQ 模型。
3. 性能：AWQ 路径在 apply 中增加了逐元素乘法操作，但仅作用于单层输入，不会显著影响整体推理性能。 - 影响：用户：使用 NVIDIA Model Optimizer 量化 NVFP4_AWQ 检查点的用户可以无需额外配置直接加载模型进行推理。系统：新增的 per-input-channel pre_quant_scale 参数会增加少量显存占用（每层 input_size_per_partition 个 float），但通常可忽略。团队：为未来支持更多 ModelOpt 量化变体（如 NVFP4_AWQ + FP8 混合精度）奠定了基础。

- 风险标记：核心路径变更，配置解析逻辑调整

关联脉络

- PR #31961 Change the FP8 per-tensor GEMM backend on SM120 to cuBLAS: 同为量化相关 PR，修改了同一文件（modelopt_quant.py）的 FP8 路径，可能涉及量化参数注册的前后一致性。
- PR #31202 Delete sgl-kernel AOT bmm_fp8, use flashinfer.bmm_fp8: 同为量化基础设施变更，涉及 FP8 内核统一，与当前 PR 的量化路径扩展有间接关联。