PR #21431 完整报告

sgl-project/sglang

[Diffusion] [AMD] Online MXFP4 and FP8 Quantization for Multimodal Generation

合并时间：2026-05-14 08:52

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/21431

执行摘要

- 一句话：为 multimodal 扩散添加在线 MXFP4/FP8 量化
- 推荐动作：本 PR 值得精读，尤其关注 量化配置与线性方法的扩展点设计（get_quant_method、packed_modules_mapping 注入），以及 跨模型传递量化参数的模式（在 FeedForward 等子模块中添加 quant_config 和 prefix 参数）。对于计划在 diffusion 模型上支持新量化后端的开发者，这是很好的参考示例。

功能与动机

PR body 指出目标是为 Z-Image-Turbo 和 Wan 2.2 等模型添加在线 MXFP4（AMD）和 FP8 量化，降低显存并加速推理。性能对比显示 transformer 尺寸减少 49-72%，峰值内存减少 29-43%，生成时间减少 5-18%，且 CLIP 分数保持稳定，证明量化对质量影响很小。

实现拆解

1. 新增 MXFP4 量化配置与线性方法：在 mxfp4.py 中定义 Mxfp4Config（继承 QuantizationConfig）和 Mxfp4LinearMethod（继承 LinearMethodBase）。初始化时条件导入 AITER 的 gemm_a4w4、shuffle_weight、dynamic_mxfp4_quant，并在 get_quant_method 中添加跳过小输出层（output_size < 256）和忽略层列表支持。
2. 修改模型定义透传量化参数：在 zimage.py 的 FeedForward 和 ZImageBlock 构造函数中添加 quant_config 和 prefix 参数，传递给子线性层，确保量化方法能递归应用到各层。同时为 ZImageTransformer2DModel 添加 packed_modules_mapping 字典，使 is_layer_skipped 能正确匹配融合层。
3. 扩展 FP8 在线量化支持：在 fp8.py 的 Fp8Config 中添加 packed_modules_mapping 参数，并修改 get_quant_method 调用 is_layer_skipped 时传入该映射，从而支持 --quantization-ignored-layers 对 FP8 路径生效。
4. 添加 CLI 与加载器配套：在 server_args.py 新增 --quantization 和 --quantization-ignored-layers 参数，更新帮助文本。在 transformer_load_utils.py 的 resolve_transformer_quant_load_spec 中，从模型类获取 packed_modules_mapping 并注入到量化配置对象中。在 fsdp_load.py 中添加 weight_scale 和 input_scale 加载键以支持量化参数。
5. Flash Attention 回退以兼容 ROCm：在 flash_attention_v3.py 中，当 sgl-kernel 的 FA3 不支持时（如 ROCm），回退到 flash_attn 包的 FA2 实现，确保 MXFP4 量化在

AMD 上也能使用 Flash Attention。

6. 更新文档：在 quantization.md 和 cli.md 中添加在线量化的使用说明、选项示例和注意事项。

关键文件：

- python/sglang/multimodal_gen/runtime/layers/quantization/mxfp4.py（模块 量化层；类别 source；类型 core-logic；符号 Mxfp4Config, Mxfp4Config.init, Mxfp4Config.get_name, Mxfp4Config.get_supported_act_dtypes）：核心新增文件，实现 MXFP4 量化配置（Mxfp4Config）与线性方法（Mxfp4LinearMethod），包括条件导入 AITER 内核、小输出层跳过、忽略层列表等关键逻辑。
- python/sglang/multimodal_gen/runtime/models/dits/zimage.py（模块 模型定义；类别 source；类型 data-contract；符号 FeedForward.init, ZImageBlock.init, ZImageTransformer2DModel.packed_modules_mapping）：Z-Image 模型适配，在 FeedForward 和 ZImageBlock 中透传 quant_config 与 prefix，使子线性层能被 MXFP4/FP8 量化；添加 packed_modules_mapping 映射支持。
- python/sglang/jit_kernel/flash_attention_v3.py（模块 注意力计算；类别 source；类型 dependency-wiring；符号 flash_attn_varlen_func）：使 Flash Attention 在 ROCm 上可用（回退到 FA2），是 AMD 平台量化推理的必要基础。
- python/sglang/multimodal_gen/runtime/server_args.py（模块 服务器配置；类别 source；类型 configuration；符号 ServerArgs, add_cli_args）：添加 --quantization 和 --quantization-ignored-layers CLI 参数，是用户控制在线量化的入口。
- python/sglang/multimodal_gen/runtime/layers/quantization/fp8.py（模块 量化器；类别 source；类型 core-logic；符号 Fp8Config.init, is_layer_skipped）：扩展 Fp8Config 以支持 packed_modules_mapping，使 --quantization-ignored-layers 在 FP8 路径中也能生效。

关键符号：Mxfp4Config.init, Mxfp4Config.get_quant_method, Mxfp4LinearMethod.create_weights, Mxfp4LinearMethod.process_weights_after_loading, Mxfp4LinearMethod.apply, FeedForward.init, Fp8Config.init, resolve_transformer_quant_load_spec, flash_attn_varlen_func

关键源码片段

python/sglang/multimodal_gen/runtime/layers/quantization/mxfp4.py

核心新增文件，实现 MXFP4 量化配置（Mxfp4Config）与线性方法（Mxfp4LinearMethod），包括条件导入 AITER 内核、小输出层跳过、忽略层列表等关键逻辑。

```
# python/sglang/multimodal_gen/runtime/layers/quantization/mxfp4.py
import logging
import torch
from sglang.srt.utils import is_hip, mxfp_supported

# 缓存 is_hip() 结果避免重复调用
_is_hip = is_hip()
```

```
if _is_hip:
    try:
        import aiter
        from aiter.ops.gemm_op_a4w4 import gemm_a4w4
        from aiter.ops.shuffle import shuffle_weight
        from aiter.utility.fp4_utils import dynamic_mxfp4_quant
    except ImportError as e:
        # 若 AITER 不可用，所有 kernel 指针置 None，后续禁用 MXFP4
        logger.warning(f"aiter MXFP4 kernels not available: {e}")
        aiter = None
        shuffle_weight = None
        dynamic_mxfp4_quant = None
        gemm_a4w4 = None

# gemm_a4w4 在输出维度 N < 256 时精度下降，因此跳过小输出层
_MXFP4_MIN_OUTPUT_DIM = 256

class Mxfp4Config(QuantizationConfig):
    """MXFP4 量化配置，适用于 diffusion 模型在线量化"""

    def __init__(self, is_checkpoint_mxfp4_serialized=False, ignored_layers=None, packed_
    modules_mapping=None):
        super().__init__()
        self.is_checkpoint_mxfp4_serialized = is_checkpoint_mxfp4_serialized
        self.ignored_layers = ignored_layers or []
        self.packed_modules_mapping = packed_modules_mapping or {}

    @classmethod
    def get_name(cls) -> str:
        return "mxfp4"

    @classmethod
    def get_min_capability(cls) -> int:
        return 95 # 对应 gfx95x，但仍建议使用 mxfp_supported() 动态判断

    def get_quant_method(self, layer, prefix: str):
        # 只量化 LinearBase 子类
        if isinstance(layer, LinearBase):
            # 若层前缀匹配忽略列表，则返回未量化方法
            if is_layer_skipped(prefix, self.ignored_layers, fused_mapping=self.packed_modules_
            mapping):
                return UnquantizedLinearMethod()
            # 输出维度 < 256 时保持全精度以避免 ASM kernel 精度问题
            output_size = getattr(layer, "output_size", None)
            if output_size is not None and output_size < _MXFP4_MIN_OUTPUT_DIM:
                return UnquantizedLinearMethod()
            return Mxfp4LinearMethod(self)
        return None
```

python/sglang/multimodal_gen/runtime/models/dits/zimage.py

Z-Image 模型适配，在 FeedForward 和 ZImageBlock 中透传 quant_config 与 prefix，使子线性层能被 MXFP4/FP8 量化；添加 packed_modules_mapping 映射支持。

```
# python/sglang/multimodal_gen/runtime/models/dits/zimage.py
class FeedForward(nn.Module):
    # 新增 quant_config 和 prefix 参数，使子线性层能接收量化配置
    def __init__(self, dim: int, hidden_dim: int,
                 quant_config: Optional[QuantizationConfig] = None,
                 prefix: str = ""):
        super().__init__()
        # 将 quant_config 和 prefix 传递给 MergedColumnParallelLinear 与 RowParallelLinear
        self.w13 = MergedColumnParallelLinear(
            dim, [hidden_dim, hidden_dim], bias=False, gather_output=False,
            quant_config=quant_config, prefix=f"{prefix}.w13")
        self.w2 = RowParallelLinear(
            hidden_dim, dim, bias=False, input_is_parallel=True,
            quant_config=quant_config, prefix=f"{prefix}.w2")
        self.act = SiluAndMul()

# ZImageTransformer2DModel 类中新增 packed_modules_mapping 静态变量
class ZImageTransformer2DModel(CachableDiT, OffloadableDiTMixin):
    packed_modules_mapping = {
        "w13": ["w1", "w3"], # 映射融合层名称，供 is_layer_skipped 正确识别
    }
```

评论区精华

- 代码风格：缓存 _is_hip 结果：mickqian 询问能否避免模块级 local 变量 _is_hip。ColinZ22 回应这是代码库惯例，类似 activation.py 和 fp8_utils.py，用于性能缓存。结论：维持原样。
- 文档覆盖：mickqian 要求更新 cli.md 和 quantization.md。ColinZ22 随后添加了对应文档，满足要求。
- FP8 路径损坏：HaiShaw 报告 main 上 FP8 路径已损坏（--quantization fp8 报错）。ColinZ22 建议在 Fp8LinearMethod.apply 上添加 @torch.compiler.disable 以规避 Inductor 不可降低 aten._scaled_mm 的问题。该修复不在本 PR 范围内，但提供了临时方案。
- quantization-ignored-layers 实际使用：avjves 在 issue 评论中指出该 CLI 参数似乎未被使用。后续提交通过 resolve_transformer_quant_load_spec 将 packed_modules_mapping 注册到量化配置，使忽略层逻辑生效。该担忧已解决。
- 代码风格：缓存 is_hip() 结果 (style): 维持原样，保留缓存变量。
- 文档更新要求 (documentation): ColinZ22 添加了对应文档，更新后满足要求。
- FP8 路径损坏与 torch.compile 兼容性 (correctness): 本 PR 未修复该问题，但提供了临时方案；后续需跟进修复。

- quantization-ignored-layers 参数未被实际使用 (correctness): 后续提交通过 resolve_transformer_quant_load_spec 从模型类获取 packed_modules_mapping 并注入量化配置，使忽略层逻辑生效。

风险与影响

- 风险：
  - 硬件依赖：MXFP4 量化依赖 AMD MI350+（gfx95x）和 AITER 库，非 AMD 平台或缺少 AITER 时将回退到未量化路径，但可能引入意料之外的 import 错误。
  - FP8 兼容性：当前 --quantization fp8 与 --enable-torch-compile 不兼容（aten._scaled_mm 无法被 Inductor 降低），虽然本 PR 未修复此问题，但用户可能遇到错误。
  - 精度衰退风险：尽管 CLIP 分数验证了图像质量，但层跳过逻辑（小输出层保持未量化）和包装层映射可能影响量化一致性，尤其当 packed_modules_mapping 不完整时。
  - 忽略层配置脆弱性：用户提供的忽略层模式依赖层前缀字符串，若模型结构更新导致前缀变化，配置可能失效。
  - 测试覆盖缺失：本 PR 未包含专门的测试文件，量化路径的可靠性依赖集成测试和 CI，存在回归风险。
- 影响：
  - 用户影响：AMD 用户可显著节省显存（MXFP4 最高 72%）并提升生成速度（21%），FP8 用户也可获得约 49% 的变压器压缩。新 CLI 参数向后兼容，不影响现有工作流。
  - 系统影响：增加对 AITER 内核的条件依赖（仅 AMD 平台加载），不影响其他硬件。GPU 内存占用降低，有利于多实例部署。
  - 团队影响：需维护 MXFP4 量化端到端链路，包括 aiter 版本兼容性。与已有 FP8 量化路径共享部分基础设施（如 is_layer_skipped），减少维护负担。
  - 风险标记：依赖 AMD MI350+ 与 AITER 库，FP8 与 torch.compile 不兼容，忽略层配置依赖层前缀字符串，缺少单元测试覆盖

关联脉络

- PR #23373 [WIP] Online MXFP4 Quantization for Diffusion: 本 PR body 提及 #23373 实现类似功能但被本 PR 的超集覆盖，建议合并到本 PR。
- PR #20922 Online FP8 quantization for multimodal generation: 为 diffusion 添加基础 FP8 在线量化，本 PR 的 FP8 部分基于该工作扩展。
- PR #26261 [AMD] Re-activate fp8 aiter backend: 修复 FP8 AITER 后端，与讨论中 HaiShaw 指出的 FP8 路径损坏问题相关。