PR #24879 完整报告

sgl-project/sglang

[AMD] support fp8 blockwise quantization combine for mori ep

合并时间：2026-05-13 14:24

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24879

执行摘要

- 一句话：支持 MoRI EP 的 FP8 blockwise 量化 combine
- 推荐动作：值得精读。展示了如何用枚举替换布尔标志提升可扩展性，以及如何与外部库协作安全引入新量化模式。尤其适合关注 AMD 平台性能优化的工程师。

功能与动机

Issue #24866 报告了在启用 FP8 combine 时 GSM8K 精度下降的问题，原因是缺少正确的量化校正。此 PR 通过集成 MoRI 的 FP8 blockwise 量化 combine 来解决，并配合 MoRI PR #311 的上游实现。

实现拆解

1. 引入枚举类型：在 moriep.py 中定义 DispatchDtype（bf16/fp8/fp4）和 CombineDtype（bf16/fp8/fp8_direct_cast），替换原有的布尔标志 fp8_dispatch、fp4_dispatch，使 dtype 配置更加可扩展且类型安全。
2. 修改 init_mori_op 函数：将参数从布尔改为枚举类型 dispatch_dtype 和 combine_dtype；在 combine dtype 为 fp8 时设置 combine_quant_type = "fp8_blockwise"，否则保持原有逻辑。
3. 环境变量支持与向后兼容：新增 SGLANG_MORI_COMBINE_DTYPE（auto/bf16/fp8/fp8_direct_cast）控制 combine dtype；统一 SGLANG_MORI_DISPATCH_DTYPE（auto/bf16/fp8/fp4）并弃用旧的 SGLANG_MORI_FP8_DISP/SGLANG_MORI_FP4_DISP；对弃用变量显示警告。
4. 块大小常量与 scale_dim 计算：在文件顶部定义 FP8_BLOCK_SIZE = 128 和 MXFP4_BLOCK_SIZE = 32；在 init_mori_op 中用这些常量计算 scale_dim，取代魔法数字。
5. Dockerfile 更新：将 docker/rocm.Dockerfile 中的 MORI_COMMIT 从 v1.1.1 更新为包含 FP8 blockwise combine 支持的特定 commit。

关键文件：

- python/sglang/srt/layers/moe/token_dispatcher/moriep.py（模块 调度器；类别 source；类型 core-logic；符号 DispatchDtype, CombineDtype）：主要实现文件，引入 DispatchDtype、CombineDtype 枚举，修改 init_mori_op 参数和环境变量处理，增加块大小常量。

- docker/rocm.Dockerfile（模块 部署脚本；类别 infra；类型 infrastructure）：更新 MoRI 版本以包含 FP8 blockwise combine kernel 支持。

关键符号：init_mori_op, CombineeDtype, DispatchDtype

关键源码片段

python/sglang/srt/layers/moe/token_dispatcher/moriep.py

主要实现文件，引入 DispatchDtype、CombineDtype 枚举，修改 init_mori_op 参数和环境变量处理，增加块大小常量。

```
# 块大小常量：每组共享一个 scale 的元素数
FP8_BLOCK_SIZE = 128
MXFP4_BLOCK_SIZE = 32


class DispatchDtype(Enum):
    """Dispatch 的量化类型枚举。"""
    bf16 = "bfloat16"
    fp8 = "float8_blockwise"
    fp4 = "mxfp4_blockwise"


class CombineDtype(Enum):
    """Combine 的量化类型枚举。"""
    bf16 = "bfloat16"
    fp8 = "float8_blockwise"
    fp8_direct_cast = "float8_direct_cast"


@lru_cache(maxsize=4)
def init_mori_op(
    group,
    router_topk,
    num_experts,
    num_local_experts,
    hidden_size,
    params_dtype,
    num_max_dispatch_tokens_per_rank,
    deepep_mode,
    instance_id=0,
    # 之前是 fp8_dispatch=False, fp4_dispatch=False
    dispatch_dtype=DispatchDtype.bf16,
    combine_dtype=CombineDtype.bf16,
    enable_sdma=False,
):
    # ... 其他代码 ...
    # 根据 dispatch_dtype 计算 scale_dim
    if dispatch_dtype == DispatchDtype.fp8:
```

```
        scale_dim = hidden_size // FP8_BLOCK_SIZE
    elif dispatch_dtype == DispatchDtype.fp4:
        # FP4 kernel 需要保持原始 hidden_size，内部做量化
        hidden_dim = hidden_size
        scale_dim = hidden_size // MXFP4_BLOCK_SIZE
        data_type = torch.float4_e2m1fn_x2
        scale_type_size = torch.float8_e8m0fnu.itemsize
    # ...
    # 处理 combine_quant_type
    combine_quant_type = "none"
    if combine_dtype == CombineDtype.fp8:
        combine_quant_type = "fp8_blockwise"
    elif combine_dtype == CombineDtype.fp8_direct_cast:
        combine_quant_type = "fp8_direct_cast"
    # ...
```

评论区精华

HaiShaw 在代码审查中要求为使用的块大小添加注释。billishyahao 回应已添加注释，并解释块大小由 MoRI 内部处理，对 SGLang 端不可见。该讨论已解决。

- 块大小注释要求 (style): billishyahao 添加了注释并解释块大小由 MoRI 内部处理，对 SGLang 不可见。

风险与影响

- 风险：
  - 回归风险：枚举替换可能导致旧的布尔参数配置失效，但提供了向后兼容的 env var 并保留弃用警告，风险可控。
  - 性能影响：根据 PR body 表格，fp8_blockwise combine 的吞吐量较 bf16 略低（如 fp4+fp8_blockwise: 784 tps vs fp4+bf16: 848 tps），但精度提高约 2%。用户需权衡速度和精度。
  - 外部依赖风险：依赖 MoRI 特定 commit，若上游更新可能需同步，但 CI 构建会验证。
  - 测试覆盖风险：缺少单元测试，仅依赖手动基准测试（16 组合的 GSM8K 精度），回归检测能力较弱。
- 影响：
  - 用户影响：AMD GPU 用户可通过环境变量选择 combine dtype，在精度敏感场景获得高达 94.5% 的 GSM8K 准确率（对比之前 ~91%）。对现有配置无破坏性变更。
  - 系统影响：改动集中在 moriep.py（+77/-36），Dockerfile 一行变更；未涉及核心推理路径或跨模块接口。
  - 团队维护成本：增加了需要跟踪 MoRI 上游的依赖，但枚举化降低了后续添加新 dtype 的复杂性。
  - 风险标记：核心路径变更，缺少测试覆盖，外部依赖变更，性能权衡

关联脉络

- 暂无明显关联 PR