PR #20520 完整报告

sgl-project/sglang

[NPU]TP Communications compression For Qwen3 models for NPU

合并时间：2026-05-02 19:29

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/20520

执行摘要

- 一句话：NPU Qwen3 TP 通信 INT8 量化，Prefill 阶段提升 5%
- 推荐动作：值得阅读以了解通信量化和推理阶段集成的设计模式。建议关注后续大规模模型上精度验证，以及是否有计划推广到其他设备（如 GPU INT8 通信）。

功能与动机

TP 通信在 attention 的 o_proj 和 MLP 的 down_proj 后进行 all_reduce，成为 prefill 阶段瓶颈。通过 INT8 量化压缩可显著降低通信延迟，实测平均性能提升 5%，且精度无损。

实现拆解

1. 在 python/sglang/srt/server_args.py 添加 --enable-quant-communications 参数。
2. 在 python/sglang/srt/distributed/device_communicators/npu_communicator.py 实现 quant_all_reduce：使用 npu_dynamic_quant 将张量量化为 INT8，然后 all_gather 量化值和缩放因子，再反量化并沿 tp 维度求和。
3. 在 python/sglang/srt/distributed/parallel_state.py 的 GroupCoordinator 中添加 quant_all_reduce 方法，当 NPU 通信器可用时委派，否则 fallback 到标准 all_reduce。
4. 在 python/sglang/srt/distributed/communication_op.py 新增 tensor_model_parallel_quant_all_reduce 和 attention_tensor_model_parallel_quant_all_reduce 函数。
5. 在 python/sglang/srt/layers/linear.py 的 RowParallelLinear.forward 中，当 forward_batch 不为 None 且 forward 模式不是 decode/idle 且参数启用时，调用量化 all_reduce。
6. 修改 python/sglang/srt/models/qwen2.py 和 qwen3.py 的 MLP forward 以传递 forward_batch。
7. 添加两个集成测试（Llama-2-7B 和 Qwen3-8B）在 GSM8K 数据集上验证准确率。
8. 更新 boolq benchmark 导入。 （文档后续单独 PR 提交）

关键文件：

- python/sglang/srt/distributed/device_communicators/npu_communicator.py（模块 NPU 通信器；类别 source；类型 core-logic；符号 quant_all_reduce, init）：核心实现：新增 quant_all_reduce 方法，使用 npu_dynamic_quant 进行 INT8 量化 all_gather 并求

和，是通信压缩的基础。

- python/sglang/srt/distributed/parallel_state.py（模块 分布式组；类别 source；类型 core-logic；符号 quant_all_reduce）：GroupCoordinator 新增 quant_all_reduce 方法，作为分布式组的唯一入口，支持 NPU 通信器委派。
- python/sglang/srt/distributed/communication_op.py（模块 通信操作；类别 source；类型 core-logic；符号 tensor_model_parallel_quant_all_reduce, attention_tensor_model_parallel_quant_all_reduce）：新增 tensor_model_parallel_quant_all_reduce 和 attention_tensor_model_parallel_quant_all_reduce 高层封装，供线性层调用。
- python/sglang/srt/layers/linear.py（模块 线性层；类别 source；类型 core-logic；符号 forward）：RowParallelLinear.forward 中根据 forward_batch 模式和服务器参数决定是否启用量化 all-reduce。
- python/sglang/srt/server_args.py（模块 启动配置；类别 source；类型 configuration）：添加 --enable-quant-communications 服务器参数及校验逻辑，作为功能的配置开关。
- python/sglang/srt/models/qwen2.py（模块 模型适配；类别 source；类型 data-contract；符号 forward）：Qwen2 MLP forward 方法新增 forward_batch 参数并传递给线性层，使量化 all-reduce 能感知推理阶段。
- python/sglang/srt/models/qwen3.py（模块 模型适配；类别 source；类型 data-contract；符号 forward）：Qwen3 MLP forward 方法类似修改，确保量化 all-reduce 在 Qwen3 上生效。
- test/registered/ascend/llm_models/test_npu_qwen3_8b_communications_quantization.py（模块 集成测试；类别 test；类型 test-coverage；符号 TestQwen38BCommQuantization）：集成测试验证 Qwen3-8B 启用量化通信后在 GSM8K 上的准确率不低于 0.85。
- test/registered/ascend/llm_models/test_npu_llama_2_7b_communications_compression.py（模块 集成测试；类别 test；类型 test-coverage；符号 TestLlama）：集成测试验证 Llama-2-7B 启用量化通信后在 GSM8K 上的准确率不低于 0.18。

关键符号：NpuCommunicator.quant_all_reduce, GroupCoordinator.quant_all_reduce, tensor_model_parallel_quant_all_reduce, attention_tensor_model_parallel_quant_all_reduce, RowParallelLinear.forward, Qwen2MLP.forward, Qwen3MLP.forward

关键源码片段

python/sglang/srt/layers/linear.py

RowParallelLinear.forward 中根据 forward_batch 模式和服务器参数决定是否启用量化 all-reduce。

```
def forward(self, input_, skip_all_reduce=False, forward_batch=None):
    # … 前面的矩阵乘法部分保持不变 …
    if self.reduce_results and self.tp_size > 1 and not skip_all_reduce:
        if self.use_dp_attention_reduce:
            output = get_attention_tp_group().all_reduce(output_parallel)
        else:
```

```
            # 决定是否启用量化通信:
            # 仅当提供了 forward_batch、且当前不是 decode 或 idle 模式、
            # 且服务器参数 enable_quant_communications 为 True 时使用 INT8 量化 all-reduce
            quantize_communications = (
                (
                    not forward_batch.forward_mode.is_decode_or_idle()
                    and get_global_server_args().enable_quant_communications
                )
                if forward_batch is not None
                else False
            )
            if quantize_communications:
                output = tensor_model_parallel_quant_all_reduce(output_parallel)
            else:
                output = tensor_model_parallel_all_reduce(output_parallel)
        else:
            output = output_parallel
        # ... 后续 bias 处理等保持不变 ...
```

评论区精华

1. 量化安全性：gemini-code-assist 指出原逻辑在 forward_batch 为 None 时可能不安全地启用量化。作者修正为仅在 forward_batch 存在且非 decode/idle 时量化。
2. 变量命名：ping1jing2 质疑 fp_comm 含义，作者改为 quantize_communications。
3. 推理阶段判断：ping1jing2 询问 is_decode() 是否应为 is_decode_or_idle()，作者采用后者。
4. 参数命名：ssshinigami 建议 --quantize-tp-communications 改为 --enable-quant-communication，作者采纳。
5. 模型架构检查：ssshinigami 建议移除仅限 Qwen3 的架构检查，作者移除并添加 Llama 测试证明通用性。

- forward_batch 为 None 时量化安全性 (correctness): 作者将逻辑改为：仅当 forward_batch 不为 None、且 forward_mode 不是 decode 或 idle、且服务器参数启用时，才使用量化 all-reduce。
- is_decode() 与 is_decode_or_idle() 的选择 (correctness): 作者改为使用 is_decode_or_idle()，确保 idle 阶段也不启用量化。
- 服务器参数命名建议 (design): 作者接受建议并修改参数名。
- 移除模型架构检查 (design): 作者移除了架构检查，并添加 Llama 测试证明通用性。

风险与影响

- 风险:
  1. 精度风险：INT8 量化引入舍入误差，但针对 GSM8K 等任务测试显示无显著下降。需要更大规模和更多任务验证，尤其对精度敏感的任务。

2. 性能风险：量化和反量化引入额外计算，可能在小 batch 或 decode 阶段带来负收益，但仅在 prefill 启用并通过 batch 大小平衡。
3. 兼容性：新参数对其他设备无影响（fallback 到标准路径）。但 NPU 特有依赖（npu_dynamic_quant）限制了可移植性。
4. 代码维护：增加了分布式通信层和模型 forward 的复杂度，需要与标准路径保持同步更新。 - 影响：对 NPU 用户：启用后 prefill 吞吐提升约 5%，无精度损失。对其他平台用户：无影响。对代码库：通信层新增量化分支，模型 forward 签名扩展，测试覆盖两个模型。整体影响范围限定在 NPU 特性和模型适配层。 - 风险标记：仅 NPU 支持，Prefill 阶段启用，INT8 量化精度风险，需更多模型覆盖

关联脉络

- 暂无明显关联 PR