# PR #33307 完整报告

sgl-project/sglang

[Perf] Broadcast single-image DP vision embedding instead of pad-to-max all-gather

合并时间：2026-08-03 10:39

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33307

## 执行摘要

- 一句话：单图 DP 视觉嵌入改 broadcast，耗时降低 7.6x
- 推荐动作：值得精读：该 PR 展示了如何用更轻量的 collective（broadcast）替换对称 all-gather，并借助手动验证脚本证明 bitwise 等价。关注点：长度计算的统一、attn_tp_group.broadcast 的抽象契约、以及单图分支的边界条件。建议后续补充跨后端（AMD/NPU/CPU）的 broadcast 兼容性测试。

## 功能与动机

PR body 指出：单图像请求时对称 all-gather 移动 tp_size x max_len 行，其中 (tp_size-1)/tp_size 是填充，每个 rank 还要分配 tp_size 倍大的缓冲区。作者希望改成从 owner rank broadcast，实现「一次拷贝而不是 tp_size 次」，并在 8x B300 上给出 7.6x 的实测加速。

## 实现拆解

1. 重构长度计算：在 python/sglang/srt/multimodal/mm_utils.py 的 run_dp_sharded_mrope_vision_model 中，将原先基于像素 patch 数的 grouped_pixel_values_len 改为基于输出 token 数的 output_tokens_per_image 和 grouped_output_lengths，使 max_len_per_rank 直接表示各 rank 输出 embedding 的实际行数，为 broadcast 路径提供准确的边界。
2. 新增单图快速路径：当 len(grid_thw_list) == 1 时，owner rank（由 image_to_tp_rank[0] 确定）直接使用 image_embeds_local.contiguous() 作为输出，其余 rank 分配 (n_tok, *shape[1:]) 空 buffer，然后调用 attn_tp_group.broadcast(out_embeddings, src=owner_local) 填充。该路径绕过了 _pad_mrope_vision_embeddings_for_tp_gather 和 all_gather，且保证多图分支完全不受影响。
3. 顺带修复CPU同步问题：packed_2d_rope分支的local_grid_thw设备改为None（CPU），避免 CUDA 上的 .tolist() 调用触发 host 同步，与广播路径的性能目标一致。
4. 测试配套：修改 test/registered/unit/models/test_kimi_k25.py，新增 test_dp_helper_broadcasts_a_single_image_from_its_owner_rank 验证单图走 broadcast 且 src 为 owner，并断言不会调用 all-gather；同时更新空 rank 场景测试以覆盖 broadcast 填充逻辑。新增 test/manual/vlm/verify_single_image_gather.py 手动验证脚本，用于在真实 NCCL 环境对比 bitwise 一致性和耗时。

5. 提交演进：4 个提交逐步完成——引入 broadcast、精简注释、覆盖 owner/receiver 双向测试、将 reproducer 移出包体放入 test/manual。

关键文件：

- python/sglang/srt/multimodal/mm_utils.py（模块 多模态；类别 source；类型 core-logic；符号 run_dp_sharded_mrope_vision_model）：核心实现文件：新增单图 broadcast 快速路径，并重构长度计算，是性能提升的关键。
- test/registered/unit/models/test_kimi_k25.py（模块 模型测试；类别 test；类型 test-coverage；符号 test_dp_helper_broadcasts_a_single_image_from_its_owner_rank, test_dp_helper_uses_config_hidden_size_for_empty_moonvit3d_rank, _GatherGroup）：新增单图 broadcast 路径的单元测试，验证 owner 源和禁止走 all-gather，保证回归安全。
- test/manual/vlm/verify_single_image_gather.py（模块 验证脚本；类别 test；类型 test-coverage；符号 main, path_a, path_b, timeit）：手动验证脚本：在真实 NCCL 环境对比 all-gather 与 broadcast 的 bitwise 一致性和耗时，是性能结论的证据来源。

关键符号：run_dp_sharded_mrope_vision_model, test_dp_helper_broadcasts_a_single_image_from_its_owner_rank, test_dp_helper_uses_config_hidden_size_for_empty_moonvit3d_rank, main

## 关键源码片段

### python/sglang/srt/multimodal/mm_utils.py

核心实现文件：新增单图 broadcast 快速路径，并重构长度计算，是性能提升的关键。

```
# 单图快速路径：单张图像只有一个 rank 持有 embedding
# 直接由 owner 广播，避免 pad-to-max all-gather 的冗余拷贝
# 条件：grid_thw_list 长度恰为 1（单个图像）
if len(grid_thw_list) == 1:
    owner_local = image_to_tp_rank[0] # owner 是负载均衡分配给该图的 rank
    n_tok = output_tokens_per_image[0] # 该图输出的 token 数（已除 embed_dim_reduction_
    factor)
    if tp_rank_local == owner_local:
        # owner rank 直接使用自己的输出，contiguous 保证 broadcast 源连续
        out_embeddings = image_embeds_local.contiguous()
    else:
        # 其他 rank 分配同样形状的空 buffer，由 broadcast 填充
        out_embeddings = torch.empty(
            (n_tok, *image_embeds_local.shape[1:]),
            dtype=input_dtype,
            device=input_device,
        )
    # 从 owner 广播，所有 rank 得到 bit-identical 结果
    get_parallel().attn_tp_group.broadcast(out_embeddings, src=owner_local)
    return out_embeddings
```

### test/registered/unit/models/test_kimi_k25.py

新增单图 broadcast 路径的单元测试，验证 owner 源和禁止走 all-gather，保证回归安全。

```
def test_dp_helper_broadcasts_a_single_image_from_its_owner_rank():
    broadcast_src = []

    class _GatherGroup:
        # 单图场景绝不允许走到 all-gather，一旦触发即失败
        def all_gather(self, tensor, dim):
            raise AssertionError("a single image must not reach the all-gather")

        # 记录 broadcast 源，并直接把输入 tensor 当作返回值（形状断言在外部）
        def broadcast(self, tensor, src):
            broadcast_src.append(src)
            return tensor

    tower = _MoonViT3dTower()
    pixel_values = torch.randn(4, 2)
    parallel = SimpleNamespace(
        attn_tp_size=2,
        attn_tp_rank=0,
        attn_tp_group=_GatherGroup(),
    )

    with patch("sglang.srt.multimodal.mm_utils.get_parallel", return_value=parallel):
        output = run_dp_sharded_mrope_vision_model(
            tower,
            pixel_values,
            [[1, 2, 2]],
            rope_type="rope_2d_packed",
        )

    assert torch.equal(output, pixel_values.reshape(1, 4, 2))
    assert broadcast_src == [0] # rank 0 是 owner
```

## test/manual/vlm/verify_single_image_gather.py

手动验证脚本：在真实 NCCL 环境对比 all-gather 与 broadcast 的 bitwise 一致性和耗时，是性能结论的证据来源。

```
def path_a(): # 旧路径：pad-to-max all-gather + 重建 owner 行
    padded = torch.empty(max_len, hidden, dtype=torch.bfloat16, device=dev)
    if emb.shape[0] > 0:
        padded[: emb.shape[0]].copy_(emb)
    gathered = [
        torch.empty(max_len, hidden, dtype=torch.bfloat16, device=dev)
        for _ in range(world)
    ]
    dist.all_gather(gathered, padded)
    return gathered[owner][:n_tok]

def path_b(): # 新路径：从 owner 广播
```

```
    buf = (
        emb.contiguous()
        if rank == owner
        else torch.empty(n_tok, hidden, dtype=torch.bfloat16, device=dev)
    )
    dist.broadcast(buf, src=owner)
    return buf

# 逐位相等性校验：A==truth, A==B, B==truth
out_a, out_b = path_a(), path_b()
eq_truth = torch.equal(out_a, owner_truth)
eq_ab = torch.equal(out_a, out_b)
eq_b_truth = torch.equal(out_b, owner_truth)
```

## 评论区精华

PR 无 review 评论，但 Issue 评论中有两次 /rerun-test 操作。首次重跑 3 个测试在 ubuntu-latest上失败，第二次扩展到5个测试后全部通过（含test_mrope_encoder_utils.py、test_kimi_k25.py、test_kimi_vl.py、test_server_args.py、test_encoder_dp.py），说明测试不稳定或存在环境依赖，最终修复通过。

- CI 重跑验证单图 broadcast 路径 (testing): 最终所有指定测试通过，包括 test_mrope_encoder_utils、test_kimi_k25、test_kimi_vl、test_server_args、test_encoder_dp。

## 风险与影响

- 风险：
  1. broadcast API 依赖：快速路径假设 attn_tp_group 一定实现 broadcast 方法，若某些后端（如自定义并行组）只实现了 all_gather，会在单图请求时崩溃。需确认所有 TP 组实现均支持。
  2. 单图判定条件：len(grid_thw_list) == 1 是进入快速路径的唯一条件，若未来出现单图但跨 rank 分片（非 DP 分配）的情况，image_to_tp_rank[0] 可能不是唯一 owner，导致广播源错误。
  3. 空图防护：当 pixel_values_local 为空且 tp_rank_local == owner_local 时，image_embeds_local 是空张量，contiguous() 后广播空 buffer，其他 rank 填充后形状依赖 image_embeds_local.shape[1:]，可能存在形状不一致。
  4. 性能回归风险：单图路径现在不再走 all-gather，若 broadcast 实现效率低于预期或驱动有 bug，可能影响单图场景的端到端时延。 - 影响：影响范围集中在多模态 DP 编码器的单图请求路径，覆盖 Kimi K25 等使用 rope_2d_packed 的模型。对多图请求和 tp_size == 1 路径无影响。实测单图 embedding 同步耗时从 0.69ms 降至 0.09ms，减少约 45MB 的跨 rank 数据移动，对高并发小图场景有明显收益。团队后续需要关注广播 API 在不同硬件（AMD、NPU）上的可用性与性能。 - 风险标记：核心路径变更，依赖 broadcast API 可用性，单图分支边界条件

# 关联脉络

- 暂无明显关联 PR