PR #7243 完整报告

verl-project/verl

[veomni] fix: preserve GPT-OSS weights without expert parallelism

合并时间：2026-08-04 07:55

原文链接：http://prhub.com.cn/verl-project/verl/pull/7243

执行摘要

- 一句话：修复 GPT-OSS 非 EP 权重导出格式错误
- 推荐动作：值得精读，重点看 get_moe_param_handler 的 " 注册表 + 显式拒绝 " 设计：对未支持路径宁可 fail loud 也不静默产生不兼容数据；以及透传 handler 与默认 handler 统一契约（name, tensor, expert_id_base）带来的可扩展性。建议后续补充 EP 与 delta_sharded 拒绝路径的测试，并将 megatron/delta_export.py 的无关格式改动拆分到独立 PR。

功能与动机

PR body 明确指出问题根源：GPT-OSS 将所有专家存放在融合 checkpoint 张量中，gate/up 值沿最后一维交错排列，而通用 VeOmni MoE 导出器按 Qwen 风格专家堆栈处理，切错维度并重命名为 per-expert 参数名。目标是为非 EP 的 GPT-OSS 选择 pass-through handler，使 vLLM 收到原始的 packed 名称、形状与值；对 EP full export 与 delta_sharded 导出则显式抛出 NotImplementedError，避免静默产出不兼容的 rank 局部表示。PR body 还专门说明与 #7195 的区别：本 PR 刻意不实现 EP 支持、不修改 vLLM，只做非 EP 全局张量的窄修复。

实现拆解

1. 新增透传 handler 并注册：在 verl/workers/engine/veomni/utils.py 中新增 passthrough_moe_param_handler(name, tensor, expert_id_base)，直接 yield name, tensor 原样返回；同时将 MOE_PARAM_HANDERS 从空字典改为 {"gpt_oss": passthrough_moe_param_handler}，使 GPT-OSS 的融合专家张量不再被 default_moe_param_handler 按 dim-0 拆分和重命名。
2. 引入统一解析器：新增 get_moe_param_handler(model_type, ep_enabled)，当 model_type 为 gpt_oss 且 ep_enabled 时抛 NotImplementedError（避免 rank 局部专家分片以 packed 全局名发送给 vLLM），其余情况回退到注册表或默认 handler。该函数成为 VeOmni 导出链路的统一入口，把 EP 判定收敛到一处。
3. 导出链路接入：verl/workers/engine/veomni/transformer_impl.py 的 get_per_tensor_param 将 process_func 的获取从 MOE_PARAM_HANDERS.get(...) 替换为 get_moe_param_handler(model_type, ps.ep_enabled)，并同步调整 import；veomni_shard_export（delta_sharded 路径）在函数入口对 gpt_oss 直接抛 NotImplementedError，避免进入不兼容的 DTensor 分片转换。

4. 测试与配套：新增 tests/utils/veomni/test_moe_param_export_on_cpu.py，参数化覆盖 gate_up_proj、down_proj 及其 bias 共 4 种形状，断言透传后输出仅 1 项、名称不变且张量为同一对象；另在 verl/workers/engine/megatron/delta_export.py 中有一处 assert 消息从多行合并为单行，属与本 PR 主题无关的格式微调。

关键文件：

- verl/workers/engine/veomni/utils.py（模块 权重导出；类别 source；类型 core-logic；符号 passthrough_moe_param_handler, get_moe_param_handler, veomni_shard_export）：核心修复所在：新增 passthrough_moe_param_handler、将 gpt_oss 注册进 MOE_PARAM_HANDERS、新增 get_moe_param_handler 解析器，并在 veomni_shard_export 入口显式拒绝 GPT-OSS 的 delta_sharded 导出。
- tests/utils/veomni/test_moe_param_export_on_cpu.py（模块 单元测试；类别 test；类型 test-coverage；符号 test_gpt_oss_non_ep_keeps_packed_expert_params）：新增参数化单元测试，直接验证非 EP 下 GPT-OSS 融合参数名与张量对象被原样透传，锁定修复行为，防止回归。
- verl/workers/engine/veomni/transformer_impl.py（模块 引擎实现；类别 source；类型 core-logic；符号 get_per_tensor_param）：导出链路接入点：get_per_tensor_param 改为通过 get_moe_param_handler 解析 handler，让 EP 判定在 full weight sync 全量导出路径中生效。
- verl/workers/engine/megatron/delta_export.py（模块 增量导出；类别 source；类型 refactor）：仅有一处 assert 消息从多行合并为单行，属于与本 PR 主题无关的格式微调，但混入了 core 文件，需在审查时注意。

关键符号：passthrough_moe_param_handler, get_moe_param_handler, veomni_shard_export, get_per_tensor_param

关键源码片段

verl/workers/engine/veomni/utils.py

核心修复所在：新增 passthrough_moe_param_handler、将 gpt_oss 注册进 MOE_PARAM_HANDERS、新增 get_moe_param_handler 解析器，并在 veomni_shard_export 入口显式拒绝 GPT-OSS 的 delta_sharded 导出。

```
# verl/workers/engine/veomni/utils.py
# 透传 handler：不拆分、不重命名，原样保留 GPT-OSS 的融合专家张量
def passthrough_moe_param_handler(name, tensor, expert_id_base):
    """Keep a fused MoE checkpoint tensor unchanged."""
    del expert_id_base
    yield name, tensor


# 按 model_type 覆盖默认 MoE 参数映射，handler 遵循
# default_moe_param_handler 契约：(name, stacked_tensor, expert_id_base)
MOE_PARAM_HANDERS = {"gpt_oss": passthrough_moe_param_handler}

def get_moe_param_handler(model_type, ep_enabled):
```

```
    """Resolve the checkpoint export handler for the current MoE layout."""
    # GPT-OSS 将所有专家存放在融合张量中，gate/up 值沿最后一维交错排列。
    # 禁用 EP 时张量已是全局形态，vLLM 期望原样保留 packed checkpoint 布局；
    # 启用 EP 时则需要 shard-aware 的 vLLM loader，绝不能用全局名称发送 rank 局部切片，
    # 因此在入口显式拒绝，而不是在同步阶段静默产出不兼容数据。
    if model_type == "gpt_oss" and ep_enabled:
        raise NotImplementedError("VeOmni GPT-OSS does not support EP weight updates")

    return MOE_PARAM_HANDERS.get(model_type, default_moe_param_handler)

# veomni_shard_export 入口：GPT-OSS 的专家权重跨 EP rank 融合存储，
# 目前没有对应的 shard-aware 转换器，delta_sharded 会生成不兼容的 rank 局部表示，
# 因此在入口显式拒绝，而不是等到同步时静默出错。
model_type = getattr(module.config, "model_type", "default")
if model_type == "gpt_oss":
    raise NotImplementedError("VeOmni GPT-OSS does not support delta_sharded weight
    updates")
```

verl/workers/engine/veomni/transformer_impl.py

导出链路接入点：get_per_tensor_param 改为通过 get_moe_param_handler 解析 handler，让 EP 判定在 full weight sync 全量导出路径中生效。

```
# verl/workers/engine/veomni/transformer_impl.py
# VeOmniEngine.get_per_tensor_param 中的 handler 解析：统一走 get_moe_param_handler，
# GPT-OSS 非 EP 命中 passthrough handler 保留 packed 权重，GPT-OSS + EP 直接抛异常。
ps = parallel_state.get_parallel_state()
model_type = getattr(self.module.config, "model_type", "default")
process_func = get_moe_param_handler(model_type, ps.ep_enabled)

def param_generator():
    for name, param in params.items():
        unsharded_tensor = param.full_tensor() if isinstance(param, DTensor) else param

        is_expert_layer = "mlp.experts." in name
        is_proj = any(p in name for p in ["down_proj", "gate_proj", "up_proj", "gate_up_proj"])

        if is_expert_layer and is_proj and ps.ep_enabled:
            # EP 路径：逐个广播各 EP rank 的局部专家块，并按全局 expert id 重命名。
            # 该路径已由 get_moe_param_handler 对 gpt_oss 显式拒绝。
            ep_rank, ep_size = ps.ep_rank, ps.ep_size
            buffer = torch.empty_like(unsharded_tensor) # [num_experts/ep_size, H, I]
            for src_ep_rank in range(ep_size):
                tensor = unsharded_tensor if src_ep_rank == ep_rank else buffer
                torch.distributed.broadcast(tensor, group_src=src_ep_rank, group=ps.ep_group)
                yield from process_func(name, tensor, expert_id_base=src_ep_rank * tensor.size(0))
        else:
            # 非 EP 路径：专家张量已全局化，expert_id_base 从 0 开始；
            # 对 GPT-OSS 而言 process_func 是 passthrough handler，不做拆分重命名。
            if is_expert_layer:
```

```
            yield from process_func(name, unsharded_tensor, expert_id_base=0)
        else:
            yield name, unsharded_tensor
```

评论区精华

本 PR 的 2 条 review 评论均来自作者 Luosuu 的自我审查，属于同一条讨论线程。核心讨论是 handler 的注册方式：作者先提出 we can register passthrough_moe_param_handler for gpt_oss in MOE_PARAM_HANDERS?，随后确认 Done. GPT-OSS now registers passthrough_moe_param_handler in MOE_PARAM_HANDERS. The resolver explicitly rejects EP updates so rank-local expert shards cannot be sent under the packed global names.。最终设计为「注册表挂 handler + 解析器显式拒绝 EP」两条职责分离；无外部 reviewer 参与，无遗留未解决问题。

- GPT-OSS 透传 handler 的注册方式 (design): 已采纳：MOE_PARAM_HANDERS = {"gpt_oss": passthrough_moe_param_handler}，由 get_moe_param_handler 统一负责 EP 拒绝逻辑，形成「注册表管映射、解析器管能力边界」的职责分离。

风险与影响

- 风险：
  - 行为断裂（有意为之）：GPT-OSS 的 EP full export 与 delta_sharded 导出从 " 静默产出错误布局 " 变为抛 NotImplementedError，存量依赖这些路径的工作流会立即失败，需要确认没有用户依赖旧行为。
  - 测试覆盖缺口：新增测试仅覆盖 CPU 单进程非 EP 透传，未覆盖 EP 拒绝路径、delta_sharded 拒绝路径，也未在 vLLM 侧做端到端加载验证。
  - 无关改动混入：verl/workers/engine/megatron/delta_export.py 的 assert 格式调整与本 PR 主题无关，增加了 review 噪音与误合并风险。
  - 隐式依赖：透传正确性依赖 vLLM 对 GPT-OSS packed 布局的既有支持，若 vLLM 侧布局认知变化，本侧没有显式信号。
- 影响：
  - 用户侧：VeOmni 引擎 + GPT-OSS + 非 EP 的 full weight sync 用户获得正确权重；EP 与 delta_sharded 用户收到显式 NotImplementedError，等待 #7195 类方案落地。
  - 系统侧：默认 handler 与 Qwen 等其他模型路径完全不受影响；变更范围集中在 verl/workers/engine/veomni/ 下两个源码文件，回归面小。
  - 团队侧：明确了 VeOmni 权重导出的支持边界（非 EP full export 支持、EP 与 delta_sharded 暂不支持），为后续 EP 方案提供了清晰的解析器契约和测试基线。
  - 风险标记：核心路径变更，EP / delta_sharded 显式报错，测试仅限 CPU 单进程，混入无关格式改动

关联脉络

- PR #7085 [veomni] feat: EP-aware sharded delta export (fused expert stacks): 改了同一批文件（verl/workers/engine/veomni/utils.py、transformer_impl.py、spec.py），是

VeOmni 权重导出体系的奠基 PR；本 PR 在非 EP 路径上修正了其通用 MoE 导出器对 GPT-OSS 融合张量的错误假设。

- PR #7181 [megatron] feat: delta_sharded on Megatron-Bridge param mappings (TP+EP, hybrid-Mamba): 涉及 verl/workers/engine/megatron/delta_export.py 与分片 delta 导出机制，与本 PR 显式拒绝 GPT-OSS delta_sharded 的决策同属 delta 导出能力边界讨论。