

PR #7161 完整报告

verl-project/verl

[fsdp] refactor: move unfuse_moe_params to FSDP backend

合并时间: 2026-07-27 23:42

原文链接: <http://prhub.com.cn/verl-project/verl/pull/7161>

执行摘要

- 一句话: 将 unfuse_moe_params 从 vLLM rollout 移至 FSDP 后端
- 推荐动作: 值得阅读, 展示了模块职责重新划分的典型手法——将 rollout 特定的转换下沉到 engine 层, 提升复用性。关注点在于移除版本检查可能带来的兼容性缺口, 建议确认 vLLM 最新版本下 per-expert 展开是否依然安全。

功能与动机

PR body 明确指出要「Move unfuse_moe_params from vllm to FSDP backend」, 让两种 rollout 后端 (vLLM / SGLang) 都能复用 MoE 参数展开逻辑, 避免重复实现。

实现拆解

1. 在 verl/workers/engine/fsdp/utils.py 新增同步生成器函数 unfuse_moe_params, 逻辑与原来的 _iter_vllm_compatible_moe_params 完全一致, 但不再依赖 asyncio。
2. 在 verl/workers/engine/fsdp/transformer_impl.py 中导入该函数, 在 get_per_tensor_param 中迭代参数后立即调用 unfuse_moe_params, 将 packed 的 MoE 专家权重展开为 per-expert 格式。
3. 从 verl/workers/rollout/vllm_rollout/vllm_rollout.py 中删除 _should_expand_vllm_moe_params 和 _iter_vllm_compatible_moe_params 两个函数, 以及 update_weights 中的相关调用条件。
4. 更新测试文件 test_vllm_moe_param_converter_on_cpu.py, 将导入改为从 FSDP utils 引入同步版本, 并移除 asyncio 包装。
5. 调整其他测试文件 (test_engine.py、test_rollout_trace_on_cpu.py 等) 以保持 CI 通过, 修复了因模块重构引起的导入或断言差异。

关键文件:

- verl/workers/engine/fsdp/utils.py (模块 FSDP 工具; 类别 source; 类型 core-logic; 符号 unfuse_moe_params): 新增核心函数 unfuse_moe_params, 是本次重构的核心输出。
- verl/workers/rollout/vllm_rollout/vllm_rollout.py (模块 vLLM rollout; 类别 source; 类型 dependency-wiring; 符号 _should_expand_vllm_moe_params, _iter_vllm_compatible_moe_params): 移除了 MoE 展开的旧函数和调用, 是重构的主要受益方, 简化了 weight update 流程。

- verl/workers/engine/fsdp/transformer_impl.py (模块 FSDP 引擎; 类别 source; 类型 dependency-wiring) : 在 get_per_tensor_param 方法中引入 unfuse_moe_params 调用, 是 FSDP 后端使用新函数的位置。
- tests/workers/rollout/test_vllm_moe_param_converter_on_cpu.py (模块 测试 /CPU; 类别 test; 类型 test-coverage; 符号 collect) : 测试文件对应更新, 从导入异步版本改为导入同步版本, 并简化测试 helper。
- tests/models/test_engine.py (模块 测试 / 引擎; 类别 test; 类型 test-coverage) : 调整了权重转换的断言逻辑以适应 engine 端处理 MoE 展开。
- tests/utis/test_rollout_trace_on_cpu.py (模块 测试 / 工具; 类别 test; 类型 test-coverage) : 修复了测试夹具中 RolloutTraceConfig 的 reset 时序问题, 避免跨测试影响。
- tests/checkpoint_engine/test_correctness_on_npu.py (模块 测试 /NPU; 类别 test; 类型 test-coverage) : 调整了 checkpoint 正确性测试中的配置, 确保与 FSDP 后端兼容。
- tests/checkpoint_engine/test_utils.py (模块 测试 / 工具; 类别 test; 类型 test-coverage) : 配合 checkpoint 引擎调整的测试更新。
- tests/workers/test_engine_return_model_output_on_cpu.py (模块 测试 / 引擎输出; 类别 test; 类型 test-coverage) : 增加一行导入以确保测试模块正确加载。
- tests/special_e2e/run_fully_async_policy.sh (模块 测试 /e2e; 类别 test; 类型 test-coverage) : 端到端测试脚本中的参数微调, 可能为适配新权重展开位置。

关键符号: unfuse_moe_params, _iter_vllm_compatible_moe_params, _should_expand_vllm_moe_params

关键源码片段

verl/workers/engine/fsdp/utis.py

新增核心函数 unfuse_moe_params, 是本次重构的核心输出。

```
# verl/workers/engine/fsdp/utis.py
```

```
def unfuse_moe_params(weights, model_type: str | None = None):
    """Expand Transformers 5 packed MoE expert tensors to vLLM checkpoint keys.

    Transformers 5 stores Qwen-style MoE experts as packed 3D parameters:
    ``mlp.experts.gate_up_proj`` with shape
    ``[num_experts, 2 * intermediate_size, hidden_size]`` and
    ``mlp.experts.down_proj`` with shape
    ``[num_experts, hidden_size, intermediate_size]``. vLLM's Qwen MoE reload
    path still accepts the original per-expert checkpoint keys during live
    weight sync, so stream those keys without materializing a full dict.
    """
    for name, tensor in weights:
        # GPT-OSS checkpoint weights use packed 3D expert tensors directly.
        # Splitting them into per-expert 2D tensors makes vLLM's GPT-OSS loader
        # index a 2D tensor as 3D during TP slicing.
```

```

if model_type == "gpt_oss" and ".mlp.experts." in name and tensor.dim() == 3:
    yield name, tensor
    continue

if name.endswith(".mlp.experts.gate_up_proj") and tensor.dim() == 3:
    gate, up = tensor.chunk(2, dim=1)
    base = name.removesuffix(".gate_up_proj")
    for expert_id in range(tensor.size(0)):
        yield f"{base}.{expert_id}.gate_proj.weight", gate[expert_id].contiguous()
        yield f"{base}.{expert_id}.up_proj.weight", up[expert_id].contiguous()
    continue

if name.endswith(".mlp.experts.down_proj") and tensor.dim() == 3:
    base = name.removesuffix(".down_proj")
    for expert_id in range(tensor.size(0)):
        yield f"{base}.{expert_id}.down_proj.weight", tensor[expert_id].contiguous()
    continue

yield name, tensor

```

verl/workers/rollout/vllm_rollout/vllm_rollout.py

移除了 MoE 展开的旧函数和调用，是重构的主要受益方，简化了 weight update 流程。

```

# verl/workers/rollout/vllm_rollout/vllm_rollout.py
# 删除前 (旧代码):
# def _should_expand_vllm_moe_params() -> bool: ...
# async def _iter_vllm_compatible_moe_params(weights, model_type): ...
# 在 update_weights 中原条件:
# if _should_expand_vllm_moe_params() and kwargs.get("peft_config") is None:
# weights = _iter_vllm_compatible_moe_params(weights, self.model_config.hf_config.model_type)

# 删除后: 上述代码全部移除, weights 直接传给 async_send_weights

```

verl/workers/engine/fsdp/transformer_impl.py

在 get_per_tensor_param 方法中引入 unfuse_moe_params 调用，是 FSDP 后端使用新函数的位置。

```

# verl/workers/engine/fsdp/transformer_impl.py
# 导入:
from .utils import create_device_mesh, get_sharding_strategy, unfuse_moe_params

# 在 get_per_tensor_param 方法中:
per_tensor_param = (
    (
        name,
        param.to(device, non_blocking=True).full_tensor() if isinstance(param, DTensor) else
        param,
    )
    for name, param in params.items()
)

```

)

注意: 移除了之前的 .to(torch.bfloat16) 转换, 因为现在假设已由外部保证 dtype

```
per_tensor_param = unfuse_moe_params(per_tensor_param, self.model_config.hf_config.model_type)
```

评论区精华

该 PR 未收到 review 评论或讨论逻辑争议。

- 暂无高价值评论线程

风险与影响

- 风险: 核心风险在于移除了 `_should_expand_vllm_moe_params` 中的版本检查 (`vLLM <= 0.24.0`), 现在 FSDP engine 中无条件展开 MoE 参数。对于 `vLLM > 0.24.0` 的版本, 直接展开是否仍兼容存在不确定性, 需要依赖上游兼容性。另外, 从异步生成器变为同步生成器, 若调用方期望异步迭代则可能出错, 但 FSDP engine 中是同步上下文, 此变更安全。影响范围涉及所有使用 FSDP engine 的 rollout (`vLLM / SGLang`), MoE 权重同步路径成为统一代码。
- 影响: 影响所有使用 FSDP engine 和 MoE 模型的训练工作流。权重同步阶段的参数格式发生了变化, 从 rollout 层处理变为 engine 层统一处理, 对用户透明, 但需要确保测试覆盖 MoE+ 非 MoE 场景。CI 中多个测试文件被连带修改, 表明该变更触发了多个下游测试的适配。
- 风险标记: 移除版本检查, 同步 / 异步语义变化, 核心权重路径

关联脉络

- PR #7109 [rollout] fix: Retain model output from tinker forward backward: 该 PR 在 `vllm_rollout.py` 的 `update_weights` 中引入了 `_iter_vllm_compatible_moe_params` 的调用, 本 PR 移除了该调用并重构到 FSDP 后端。
- PR #7158 [rollout] fix: TypeError merging numpy routed_experts on partial rollout resume: 同为 MoE 相关修复, 涉及 LLM server 对 `routed_experts` 的处理, 间接关联 MoE 权重转换路径。
- PR #7144 [ckpt, fsdp] feat: sharded delta block placements + backend-owned HF export (engine core + FSDP): 涉及 FSDP 引擎的 checkpoint 和 delta 同步, 与 FSDP 后端的权重导出逻辑有重叠。