PR #7117 完整报告

verl-project/verl

[ckpt] fix: save base model's code, not the PeftModel wrapper's, in FSDP checkpoints

合并时间：2026-08-03 15:34

原文链接：http://prhub.com.cn/verl-project/verl/pull/7117

执行摘要

- 一句话：修复 FSDP 保存 LoRA 模型时错误打包 PeftModel 源码的 bug
- 推荐动作：值得精读。虽然改动很小，但 PR body 对 transformers custom_object_save 机制和 FSDP1/FSDP2 差异失败模式的分析很有价值；测试设计（stub 分布式原语 + 负向控制验证）也值得借鉴。建议关注 get_base_model() 的 peft-only 判断，以及 save_checkpoint 中 auto_map 分支的容错方式。

功能与动机

PR body 中指出：custom_object_save 复制 type(obj).__module__的源码使 checkpoint 自包含，因此必须传入基础模型。PEFT 运行时传入的是 PeftModel wrapper，导致打包了错误模块的源码。FSDP1 下 unwrap_model 是 PeftModel，walking peft/peft_model.py 的 from . tuners import ... 解析到不存在的 peft/tuners.py，保存直接抛 FileNotFoundError；FSDP2 下类被原地替换，walk 解析到 torch 的 fsdp.py 文件存在，保存静默成功，但 auto_map 指向的 modeling_.py 从未被复制，checkpoint 无法通过 trust_remote_code 重新加载。修复通过 get_base_model() 解包，peft-only 逻辑保证非 PEFT 模型行为不变。

实现拆解

1. 定位问题根源：分析 custom_object_save 在 transformers 中的实现，它通过 type(obj).__module__ 定位并打包模型源码文件，因此必须收到基础模型对象而非 PeftModel wrapper。
2. 修改 verl/utils/checkpoint/fsdp_checkpoint_manager.py 中 save_checkpoint 的 HF 配置导出分支：在调用 custom_object_save 前用 hasattr(unwrap_model, "get_base_model") 判断是否为 PEFT 模型，是则先解包，否则原样透传。同时补充注释说明 FSDP1/FSDP2 下的两种失败模式，帮助后续维护者理解为何必须解包。
3. 新增 tests/utils/ckpt/test_lora_custom_object_save_on_cpu.py：构建最小 Qwen3 LoRA 模型，设置 config.auto_map 走 trust_remote_code 分支，mock 掉 custom_object_save 后调用 save_checkpoint，断言传入的是 get_base_model() 的返回值且不是 wrapper；并 stub 了 torch.distributed.get_rank/get_world_size/barrier 以保证单进程 CPU 下可安全运行。
4. 测试有效性验证：PR body 中提到，若将修复 revert 为直接传 unwrap_model，测试会失败，说明该测试真实守护了此行为。

关键文件：

- verl/utils/checkpoint/fsdp_checkpoint_manager.py（模块 检查点；类别 source；类型 core-logic；符号 save_checkpoint）：核心修复文件：在 save_checkpoint 的 auto_map 分支中解包 PeftModel，确保 custom_object_save 打包基础模型源码，修复 FSDP1 保存失败与 FSDP2 静默损坏问题。
- tests/utils/ckpt/test_lora_custom_object_save_on_cpu.py（模块 单测；类别 test；类型 test-coverage；符号 _make_peft_model, test_custom_object_save_receives_base_model_not_peft_wrapper）：新增 CPU 单测，首次覆盖 auto_map 保存分支，使用 mock 验证 custom_object_save 收到的是基础模型而非 wrapper，并通过负向控制验证测试有效性。

关键符号：save_checkpoint, test_custom_object_save_receives_base_model_not_peft_wrapper, _make_peft_model

关键源码片段

verl/utils/checkpoint/fsdp_checkpoint_manager.py

核心修复文件：在 save_checkpoint 的 auto_map 分支中解包 PeftModel，确保 custom_object_save 打包基础模型源码，修复 FSDP1 保存失败与 FSDP2 静默损坏问题。

```
# 保存 HF 配置与 tokenizer 到 checkpoint 的 huggingface 子目录
hf_config_tokenizer_path = os.path.join(local_path, "huggingface")
local_mkdir_safe(hf_config_tokenizer_path)

model_config = unwrap_model.config
generation_config = None

# 若模型可生成且带有 name_or_path，则尝试保存 generation config
if unwrap_model.can_generate() and hasattr(model_config, "name_or_path") and model_config.
name_or_path:
    try:
        generation_config = GenerationConfig.from_pretrained(model_config.name_or_path)
        generation_config.save_pretrained(hf_config_tokenizer_path)
    except Exception:
        # 有些模型未从预训练初始化，name_or_path 为空，跳过保存
        pass

# 清理 auto_map 中的 None 键（transformers trust_remote_code 需要）
if hasattr(model_config, "auto_map") and None in model_config.auto_map:
    model_config.auto_map = {k: v for k, v in model_config.auto_map.items() if k is not None}

model_config.save_pretrained(hf_config_tokenizer_path)
if self.processing_class is not None:
    self.processing_class.save_pretrained(hf_config_tokenizer_path)

# custom_object_save 会复制 type(obj).__module__ 的源码，使 checkpoint 自包含。
# 对 PEFT 模型，unwrap_model 是 PeftModel wrapper，若直接传它会导致：
# - FSDP1: 走 peft/peft_model.py 的 from .tuners import ...，解析到不存在的 peft/tuners.py，抛
```

```
FileNotFoundError
# - FSDP2: 走 torch 的 _fsdp_*.py，保存成功但 auto_map 指向的 modeling_*.py
未复制，checkpoint 无法加载
# 因此先通过 get_base_model() 解包到基础模型（peft-only API，非 PEFT 模型直接透传）。
if hasattr(model_config, "auto_map"):
    save_obj = unwrap_model.get_base_model() if hasattr(unwrap_model, "get_base_model") else
    unwrap_model
    custom_object_save(save_obj, hf_config_tokenizer_path, config=model_config)
```

tests/utils/ckpt/test_lora_custom_object_save_on_cpu.py

新增 CPU 单测，首次覆盖 auto_map 保存分支，使用 mock 验证 custom_object_save 收到的是基础模型而非 wrapper，并通过负向控制验证测试有效性。

```
def test_custom_object_save_receives_base_model_not_peft_wrapper(monkeypatch, tmp_path):
    """在 LoRA 保存时，传给 custom_object_save 的必须是基础模型而不是 PeftModel wrapper。"""
    # 未初始化的 process group 下安全执行：save_checkpoint 会查询 rank/world_size 并调用
    barrier()。
    # 普通 PeftModel 的 fsdp_version == 0，sharded-state-dict 分支是
    nullcontext，不会触发集合通信，
    # 因此只需 stub 三个分布式原语即可。
    monkeypatch.setattr(torch.distributed, "get_rank", lambda: 0)
    monkeypatch.setattr(torch.distributed, "get_world_size", lambda: 1)
    monkeypatch.setattr(torch.distributed, "barrier", lambda: None)

    peft_model = _make_peft_model()
    # 设置 auto_map 使代码走 trust_remote_code 分支（dict 中无 None 键，等价真实配置）。
    peft_model.config.auto_map = {"AutoModelForCausalLM": "modeling_x.Foo"}

    # save_contents=["model"] 只保存模型，跳过优化器与 HF 导出，保持测试轻量。
    ckpt_config = CheckpointConfig(save_contents=["model"], load_contents=["model"], async_
    save=False)
    manager = FSDPCheckpointManager(
        model=peft_model,
        optimizer=None,
        lr_scheduler=None,
        processing_class=None,
        checkpoint_config=ckpt_config,
    )

    mock_custom_object_save = MagicMock()
    monkeypatch.setattr("verl.utils.checkpoint.fsdp_checkpoint_manager.custom_object_save",
    mock_custom_object_save)

    manager.save_checkpoint(local_path=str(tmp_path), global_step=0)

    mock_custom_object_save.assert_called_once()
    saved_obj = mock_custom_object_save.call_args.args[0]
    assert saved_obj is peft_model.get_base_model(), "custom_object_save 必须收到基础模型"
    assert saved_obj is not peft_model, "custom_object_save 不能收到 PeftModel wrapper"
```

评论区精华

该 PR 没有技术性的 review 评论，tardis-key 直接批准。作者在评论中补充了 CI 状态说明：3 个 Ascend E2E job 被取消，但非必需且不会覆盖本 PR（Ascend workflow 对 fsdp_checkpoint_manager.py 做了 path-exclude），因此不影响合并。

- CI 状态确认 (other): reviewer tardis-key 已批准，作者请求维护者合并。

风险与影响

- 风险：回归风险低：修改仅位于 save_checkpoint 的 auto_map 分支，非 PEFT 模型因 hasattr(..., "get_base_model") 为 False 而原样透传，行为不变。依赖 PEFT 的 get_base_model() API，若该 API 行为变更可能影响修复，但测试对此有守护。测试仅在 CPU 环境使用 mock，未覆盖真实 FSDP1/FSDP2 分布式场景，不过 PR body 中已说明手动验证 get_base_model() 在两种 FSDP 版本下均返回真实基础模型。
- 影响：影响用户：使用 trust_remote_code 模型 + LoRA/PEFT + FSDP 的用户，修复前 checkpoint 可能保存失败（FSDP1）或静默生成不可加载的 checkpoint（FSDP2）；修复后 checkpoint 自包含、可恢复。影响范围小：非 PEFT 模型行为完全不变，仅 checkpoint 保存路径受影响，不涉及训练前向与推理。对团队而言这是一个单 commit、已批准的小型 bugfix。
- 风险标记：PEFT 模型分支变更，仅覆盖 CPU 测试，影响 trust_remote_code checkpoint 可加载性，依赖 peft get_base_model API

关联脉络

- PR #7161 [fsdp] refactor: move unfuse_moe_params to FSDP backend: 同为 FSDP 后端相关改动，FSDP 后端重构是 checkpoint 保存路径稳定化的背景之一。
- PR #7193 [ckpt, model] fix: validate model merger outputs: 同一 checkpoint 质量线上，关注 checkpoint 输出的可加载性与完整性。