PR #1867 完整报告

THUDM/slime

[docker] upgrade megatron to 1dcf0dafa

合并时间：2026-04-28 14:07

原文链接：http://prhub.com.cn/THUDM/slime/pull/1867

执行摘要

- 一句话：升级 Megatron 依赖至 1dcf0dafa，适配 API 变更
- 推荐动作：建议关注 model_provider.py 中的新 get_model_provider_func 设计，它解耦了模型提供和 freeze 包装，使调用更简洁。同时，wrap_model_provider_with_freeze 的 **kwargs 改造值得学习，可提高对上游签名变化的适应性。此外，确认所有使用 wrap_model_provider_with_freeze 的地方均已替换为新入口。

功能与动机

PR 未关联具体 Issue，但 body 中感谢了 radixark 团队的工作，说明此次升级是为了采纳 radixark/Megatron-LM 的改进。上游更新可能包含新功能、性能优化或关键 bugfix。

实现拆解

1. 升级 Docker 基础镜像的 Megatron 版本：修改 docker/Dockerfile 中的 MEGATRON_COMMIT 变量，将引用从旧 commit 更新到 1dcf0dafa，同时更新 docker/version.txt 记录对应版本。
2. 重新生成 Megatron 补丁：docker/patch/latest/megatron.patch 在旧补丁基础上针对新版 Megatron 的代码结构调整了 diff 上下文，保留了相同的修改意图（如 checkpoint 加载容错、parallel_mode 注入、_FakeInt4QuantizationSTE 等），确保补丁能正确应用到新版本上。
3. 重构模型提供函数接口：将原有的 get_model_provider_func 拆分为 _get_model_provider_func（内部实现）和新的 get_model_provider_func（对外入口），并将 wrap_model_provider_with_freeze 的包装逻辑内嵌到新入口中。wrap_model_provider_with_freeze 内部改用 **kwargs 可变参数并显式提取 vp_stage、config、pg_collection 等关键字，提高了对上游签名变化的鲁棒性。
4. 简化模型初始化调用：在 model.py 的 setup_model_and_optimizer 中，不再手动调用 wrap_model_provider_with_freeze，而是直接使用新的 get_model_provider_func，该函数内部已包含 freeze 包装，调用层更加简洁。
5. 增强参数收集的断言容忍度：在 update_weight/common.py 的 all_gather_param 中，将 partition_stride == 1 的硬断言放宽为允许 partition_stride == 2（仅限 linear_fc1），以适配新版 Megatron 对 GLU 层的分区方式。

6. 补充配置映射：在 arguments.py 中添加了 rms_norm_eps 到 layernorm_epsilon 的映射，并修复了未同时检查 hf_config 和 args 属性的 bug。
7. 无测试文件变更：本次升级未新增或修改测试，主要通过 CI 镜像构建验证兼容性。

关键文件：

- slime/backends/megatron_utils/model_provider.py（模块 模型提供；类别 source；类型 data-contract；符号 get_model_provider_func, _get_model_provider_func, wrapped_provider）：核心变更文件，重构了模型提供函数接口，拆分内部实现和对外入口，改变 freeze 包装方式。
- docker/patch/latest/megatron.patch（模块 Megatron 补丁；类别 test；类型 test-coverage）：更新了 Megatron 补丁以匹配新版本代码，包含 checkpoint 加载容错、parallel_mode 注入、FakeInt4 量化等修改。
- slime/backends/megatron_utils/model.py（模块 模型管理；类别 source；类型 data-contract）：简化了模型初始化调用，移除了手动 freeze 包装，直接使用新的 get_model_provider_func。
- slime/backends/megatron_utils/update_weight/common.py（模块 权重更新；类别 source；类型 core-logic）：放宽了 partition_stride 断言，适应新版本 Megatron 的 GLU 层分区方式。
- slime/backends/megatron_utils/arguments.py（模块 配置参数；类别 source；类型 core-logic）：添加了 rms_norm_eps 到 layernorm_epsilon 的映射，并修复了条件检查的防御性。
- docker/Dockerfile（模块 Docker 构建；类别 infra；类型 infrastructure）：升级 Megatron 版本 commit，驱动整个变更。
- docker/version.txt（模块 版本管理；类别 docs；类型 documentation）：记录版本号，便于追溯。

关键符号：get_model_provider_func, _get_model_provider_func, wrapped_provider, freeze_model_params, all_gather_param, setup_model_and_optimizer

关键源码片段

slime/backends/megatron_utils/model_provider.py

核心变更文件，重构了模型提供函数接口，拆分内部实现和对外入口，改变 freeze 包装方式。

```
# slime/backends/megatron_utils/model_provider.py

def _get_model_provider_func(
    args: argparse.Namespace,
    role: Literal["actor", "critic"] = "actor",
):
    """内部实现：构建模型 provider 函数（不包含 freeze 包装）"""
    # ... 原有逻辑 ...


def wrap_model_provider_with_freeze(original_provider, args):
```

```
    def wrapped_provider(
        pre_process=True,
        post_process=True,
        **kwargs, # 改为可变参数，灵活接收上游可能新增的关键字
    ):
        sig = inspect.signature(original_provider)
        provider_kwargs = {
            "pre_process": pre_process,
            "post_process": post_process,
        }
        # 显式提取已知的关键字参数，忽略未知的
        for key in ["vp_stage", "config", "pg_collection"]:
            if key in sig.parameters:
                provider_kwargs[key] = kwargs.get(key, None)

        model = original_provider(**provider_kwargs)
        freeze_model_params(model, args)
        return model

    return wrapped_provider


def get_model_provider_func(args, role="actor"):
    """对外统一入口：将 freeze 包装与模型提供组合，调用方无需再手动 wrap"""
    return wrap_model_provider_with_freeze(_get_model_provider_func(args, role), args)


def freeze_model_params(model: GPTModel, args: argparse.Namespace):
    # 改用 getattr 安全访问，避免缺少属性时报错
    if getattr(args, "only_train_params_name_list", None):
        # ...
    if getattr(args, "freeze_params_name_list", None):
        # ...
```

评论区精华

本 PR 没有收集到 review 讨论。由于作者同时是合并者，变更可能经过内部评审后快速合并。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 兼容性风险：新版 Megatron 可能引入未预料到的 API 变更，影响模型训练、checkpoint 加载等核心路径。尽管补丁和源码已做适配，但未增加覆盖新版本的单元测试或集成测试。
  2. 回归风险：model_provider.py 的重构改变了函数调用链和参数传递方式，可能影响到依赖 wrap_model_provider_with_freeze 的外部调用（如 train.py 等未在此 PR 中更新的

文件）。需要确认所有调用点都已迁移到新接口。

3. 性能风险：_FakeInt4QuantizationSTE 仍然保留，但新版本 TE 的行为可能有所变化。
4. 构建风险：Docker 镜像依赖指定 commit，若该 commit 被强制推送或不可用，未来构建可能失败。 - 影响：用户：需要重新构建 Docker 镜像才能使用新版本 Megatron；已训练的 checkpoint 在加载时可能因版本差异需要新的容错逻辑（补丁已包含）。系统：训练流程中的模型初始化、参数收集和配置校验将使用新版 Megatron 的实现，可能带来细微的行为差异。团队：需关注后续 Megatron 版本演进，确保补丁和适配代码能持续同步。

- 风险标记：核心路径变更，缺少测试覆盖，兼容性风险，配置映射调整

关联脉络

- PR #1859 [docker] cleanup sglang patch: 都涉及 Docker 镜像的补丁维护，清理 sglang 补丁与升级 megatron 补丁属于同一基础设施维护线。
- PR #1866 Rename critic config to megatron config: 都影响了 slime/backends/megatron_utils 下的配置和 API，存在重叠文件。
- PR #1856 refactor/ppo: PPO 架构重构中大量涉及 model_provider 和 model.py 的改动，本次升级需与之兼容。