

PR #3183 完整报告

radixark/miles

[AMD][LoRA] Add a Qwen3-4B LoRA RL launcher

合并时间: 2026-09-15 05:33

原文链接: <http://prhub.com.cn/radixark/miles/pull/3183>

执行摘要

- 一句话: 补齐 AMD 平台 Qwen3-4B LoRA RL 启动器
- 推荐动作: 值得精读。若团队需要维护或扩展 AMD / ROCm 上的 LoRA RL 训练, 这是当前唯一入口; 其「平台差异收敛在启动器层」的做法——环境变量桥接 (HIP → CUDA 可见设备)、SGLang Triton 后端固定、以及「配方 + 快照 + 文档」三件套的组织方式——是值得复用的模式。验证数据 (logprob 差、KL、截断率、AIME 趋势) 可作为后续 AMD 训练健康度对比的参考基线, 但不应据此断言训练收益。

功能与动机

PRbody开头即点明动机: `ThereisnomaintainedQwen3-4BLoRARLlauncherforAMD`。

此前 AMD 用户只能从 CUDA 配方自行移植, 缺少可维护、可复现的入口; 同时 ROCm 上 SGLang 的 CUDA-only rollout 后端不可用, 必须在启动器层把 attention 与 LoRA 后端固定为 Triton 并做 HIP/CUDA 设备环境桥接, 才能把已验证的 Megatron-Bridge LoRA 配方带到 AMD 平台。

实现拆解

1. 新增启动器入口: `scripts/amd/run_qwen3_4b_lora.py` 定义 `ScriptArgs` (继承 `U.ExecuteTrainConfig`), 以 `dataclass` 声明全部可调项: `hardware` 限定为 `auto` / `MI350X` / `MI355X`, `num_gpus_per_node` 允许覆盖可见卡数; LoRA 超参 (`rank 64`、`alpha 32`、`dropout 0.0`、`target-modules all-linear`、`lora-base-cpu-backup`) 沿用已验证的 Qwen3-4B Megatron LoRA 配方, rollout 形状 (`8 prompt × 8 样本 = 64 全局 batch`) 与 `8192 token` 响应窗口对齐既有 Qwen3 launcher。
2. ROCm 环境适配: `_set_rocm_environment()` 设置 `RAY_EXPERIMENTAL_NOSET_HIP_VISIBLE_DEVICES` 与 `RAY_EXPERIMENTAL_NOSET_CUDA_VISIBLE_DEVICES`, 阻止 Ray 启动 worker 时重设设备可见性; 把 `HIP_VISIBLE_DEVICES` 同步到 `CUDA_VISIBLE_DEVICES`, 保证上层按 CUDA 习惯读取设备; 再设 `NCCL_NVLS_ENABLE=0` 避开 `execute_train` 的 NVIDIA 拓扑探测与 ROCm 不支持的 NVLink SHARP。
3. 命令组装与提交: `_execute()` 按 `checkpoint` / `LoRA` / `rollout` / `eval` / `optimizer` / `GRPO` / `SGLang` / `misc` 八组参数块拼接训练命令, 其中 SGLang 块固定 `--sglang-lora-backend triton` 与 `--sglang-attention-backend triton`, 并设置 `--sglang-chunked-prefill-size 4096`、`--sglang-mem-fraction-static 0.4`; 最终调用 `U.execute_train` 提交 Ray job, `--colocate`

让 actor 与 rollout 共享 8 卡。

4. CLI 入口划分：prepare 只下载模型（Qwen/Qwen3-4B）与数据集（dapo-math-17k、可选 aime-2024）；train 假设输入已在节点上直接训练；full-train 合并两者，便于一键跑通。
5. 测试与文档配套：tests/manual/launch_scripts/test_py_launch_scripts.py 把新脚本加入 `_HARDWARE_A_RECORDING_REPRESENTS`（固定 MI355X）；新增 tests/snapshots/launch_scripts/py/scripts/amd/run_qwen3_4b_lora.py/ 下 prepare / train / full_train 三个命令录制快照，作为启动器行为回归基准；docs/advanced/lora.md 的 LoRA 支持矩阵补充 AMD 启动器条目并注明 Triton 后端。作者在 8x MI355X、ROCm 7.2、torch 2.9.1+rocm7.2.0 上完成 100 个 optimizer 步骤，train/train_rollout_logprob_abs_diff 均值 0.01232（低于 0.03 门限），AIME-2024 avg@16 从 0.5625 提升至峰值 0.6833。

关键文件：

- scripts/amd/run_qwen3_4b_lora.py（模块 启动器；类别 source；类型 core-logic；符号 ScriptArgs, _set_rocm_environment, _resolve_num_gpus, _download_inputs）：本 PR 的核心：新增 Qwen3-4B GRPO LoRA 启动器，集中处理 AMD 硬件声明、ROCm 环境变量桥接、SGLang Triton 后端固定与训练命令组装，是 AMD LoRA RL 的维护入口。
- tests/manual/launch_scripts/test_py_launch_scripts.py（模块 启动测试；类别 test；类型 test-coverage；符号 `_HARDWARE_A_RECORDING_REPRESENTS`）：把新启动器注册到 `_HARDWARE_A_RECORDING_REPRESENTS`（固定为 MI355X），让快照测试以正确的硬件假设录制命令序列。
- tests/snapshots/launch_scripts/py/scripts/amd/run_qwen3_4b_lora.py/full_train.txt（模块 命令快照；类别 test；类型 test-fixture）：full-train 入口的完整命令录制快照（109 行），覆盖 HF 下载、Ray 启动与 train.py 全量参数，是启动器行为回归的主要依据。
- tests/snapshots/launch_scripts/py/scripts/amd/run_qwen3_4b_lora.py/train.txt（模块 命令快照；类别 test；类型 test-fixture）：train 入口的命令录制快照（92 行），锁定训练参数序列，防止默认值漂移导致启动器行为悄悄变化。
- tests/snapshots/launch_scripts/py/scripts/amd/run_qwen3_4b_lora.py/prepare.txt（模块 命令快照；类别 test；类型 test-fixture）：prepare 入口的下载命令录制快照（16 行），校验模型与数据集下载路径与仓库选择。
- docs/advanced/lora.md（模块 文档；类别 docs；类型 documentation）：LoRA 支持矩阵补充 AMD 启动器条目并注明 Triton attention/LoRA 后端，让用户能直接定位 AMD 支持状态。

关键符号：ScriptArgs, _set_rocm_environment, _resolve_num_gpus, _download_inputs, _get_wandb_args, _execute, prepare, train, full_train

关键源码片段

scripts/amd/run_qwen3_4b_lora.py

本 PR 的核心：新增 Qwen3-4B GRPO LoRA 启动器，集中处理 AMD 硬件声明、ROCm 环境变量桥接、SGLang Triton 后端固定与训练命令组装，是 AMD LoRA RL 的维护入口。

scripts/amd/run_qwen3_4b_lora.py（新增启动器核心，节选）

```
# 平台差异被收敛在启动器层：参数声明、ROCm 环境适配与命令组装都在这里完成，
# 避免把 AMD 特例写进核心训练代码。
```

```
@dataclass
```

```
class ScriptArgs(U.ExecuteTrainConfig):
```

```
    hardware: Literal["auto", "MI350X", "MI355X"] = "auto"
```

```
    num_gpus_per_node: int | None = None
```

```
    # LoRA 超参沿用已在 CUDA 上验证的 Qwen3-4B Megatron LoRA 配方
```

```
    lora_rank: int = 64
```

```
    lora_alpha: int = 32
```

```
    lora_dropout: float = 0.0
```

```
    target_modules: str = "all-linear"
```

```
    lora_base_cpu_backup: bool = True
```

```
    # Rollout 形状：8 个 prompt × 8 个样本 = 一次 64 样本更新
```

```
    num_rollout: int = 100
```

```
    rollout_batch_size: int = 8
```

```
    n_samples_per_prompt: int = 8
```

```
    # 8192 token 响应窗口与已维护的 Qwen3 launcher 对齐
```

```
    rollout_max_response_len: int = 8192
```

```
    global_batch_size: int = 64
```

```
    lr: float = 2e-5
```

```
    # 每个 GPU 一个 4B rollout engine
```

```
    rollout_num_gpus_per_engine: int = 1
```

```
    slang_mem_fraction_static: float = 0.4
```

```
def _set_rocm_environment() -> None:
```

```
    """在 Ray 启动前后保持调用方的 ROCm 设备选择。"""
```

```
    # 阻止 Ray 在启动 worker 时重设 HIP/CUDA 可见设备
```

```
    os.environ.setdefault("RAY_EXPERIMENTAL_NOSET_HIP_VISIBLE_DEVICES", "1")
```

```
    os.environ.setdefault("RAY_EXPERIMENTAL_NOSET_CUDA_VISIBLE_DEVICES", "1")
```

```
    # 把 HIP_VISIBLE_DEVICES 同步到 CUDA_VISIBLE_DEVICES,
```

```
    # 让上层逻辑仍按 CUDA 习惯读取设备编号
```

```
    if hip_visible_devices := os.environ.get("HIP_VISIBLE_DEVICES"):
```

```
        os.environ["CUDA_VISIBLE_DEVICES"] = hip_visible_devices
```

```
    # execute_train 会触发 NVIDIA 拓扑探测；ROCm 上禁用不支持的 NVLink SHARP
```

```
    os.environ.setdefault("NCCL_NVLS_ENABLE", "0")
```

```
def _execute(args: ScriptArgs) -> None:
```

```
    _set_rocm_environment()
```

```
    hardware, num_gpus = _resolve_num_gpus(args)
```

```
    # LoRA 参数块：lora-base-cpu-backup 让基座权重常驻 CPU，缓解显存压力
```

```
    lora_args = (
```

```
        f"--lora-rank {args.lora_rank} "
```

```

f"--lora-alpha {args.lora_alpha} "
f"--lora-dropout {args.lora_dropout} "
f"--target-modules \"{args.target_modules}\" '
)
if args.lora_base_cpu_backup:
    lora_args += "--lora-base-cpu-backup "

# 本启动器与 CUDA 配方最大的差异点: ROCm 没有 CUDA-only 的 SGLang rollout 后端,
# 必须把 attention 与 LoRA 后端固定为 Triton
sglang_args = (
    f"--rollout-num-gpus-per-engine {args.rollout_num_gpus_per_engine} "
    f"--sglang-mem-fraction-static {args.sglang_mem_fraction_static} "
    "--sglang-chunked-prefill-size 4096 "
    "--sglang-lora-backend triton "
    "--sglang-attention-backend triton "
)

# 单节点 colocated: actor 与 rollout 共享 8 卡, 训练与采样在同一 Ray 集群内完成
misc_args = (
    "--train-backend megatron "
    f"--actor-num-nodes {args.num_nodes} "
    f"--actor-num-gpus-per-node {num_gpus} "
    f"--num-gpus-per-node {num_gpus} "
    "--colocate "
)

# 其余参数块 (checkpoint / rollout / eval / optimizer / GRPO / wandb) 在此省略,
# 最终统一交给通用命令工具层提交 Ray job; config 对象与参数字符串共同描述一次训练
train_args = " ".join(
    [ckpt_args, lora_args, rollout_args, eval_args, optimizer_args,
     grpo_args, _get_wandb_args(args), sglang_args, misc_args, args.extra_args]
)

U.execute_train(
    train_args=train_args,
    config=args,
    num_gpus_per_node=num_gpus,
    megatron_model_type=args.megatron_model_type,
    megatron_path=args.megatron_path,
)

```

评论区精华

维护者 yushengsu-thu 在 review 中提问 [Should this PR add snapshots?](#) (指向 @yueming-yuan), 关注新启动器是否应随命令录制快照一起合入; 最终版本已包含 prepare / train / full_train 三个快照文件, 且 yushengsu-thu 随后给出 APPROVED, 说明该疑问在合并前得到解决。另 claude[bot] 提示本 PR 来自 fork、自动 review 被禁用, 维护者可评论 [@claude review](#) 触发一次性 review; 实际未触发, 由维护者人肉审核并批准。

- 是否应把启动器快照纳入本 PR (question): 最终版本已包含 prepare.txt、train.txt、full_train.txt 三个快照文件，且 yushengsu-thu 随后给出 APPROVED，说明该疑问在合并前得到解决。
- fork PR 自动 review 被禁用 (other): 未触发额外 bot review，由维护者 yushengsu-thu 人肉审核并批准合并。

风险与影响

- 风险：
 1. 快照对工具链版本敏感：tests/snapshots 冻结了极长且细节级的 train.py 参数序列（含全部模型超参与 --wandb-key 'frozen-wandb-api-key'），一旦 train.py 默认参数演进或 SGLang 参数改名，快照即失效，需要同步更新。
 2. ROCm 环境变量依赖：_set_rocm_environment() 依赖 Ray 的 RAY_EXPERIMENTAL_NOSET_* 行为与 SGLang 在 ROCm 上的设备读取约定，相关版本升级后可能失效且难以在 CPU CI 捕获。
 3. 验证范围有限：仅 8x MI355X 单节点 colocated 实测，MI350X 与多节点 / 跨节点场景、ROCm 7.2 之外版本未覆盖；colocated 模式下 actor 与 rollout 同机显存竞争的行为未展开讨论。
 4. 评估信号强度不足：AIME-2024 只有 30 道独立题目，avg@16 的涨跌统计功效有限，作者自己也声明这是次要信号而非效果声明。
 5. 无 GPU CI 覆盖：该启动器只进入 tests/manual 层，PR CI 不实际跑 ROCm 训练，回归只能依赖快照与人工验证。- 影响：对 AMD / ROCm 用户，这是开箱即用的 Qwen3-4B LoRA RL 起点，prepare / train / full-train 三步 CLI 大幅降低上手成本；对系统，核心训练库零改动（+469/-1 中仅 1 行删除为文档表格），训练路径无回归面，新增快照扩大了 launcher 回归保护；对团队，填补 AMD launcher 维护空白，后续 AMD LoRA 迭代可在该脚本上延续，LoRA 支持矩阵也让用户能直接定位 AMD 状态。整体影响范围为中低，聚焦新增入口与配套资产。- 风险标记：ROCm 环境变量依赖，快照对工具链版本敏感，新入口无 GPU CI 覆盖，仅单机 colocated 验证，AIME 评估样本量小

关联脉络

- PR #2701 feat(ci): declare supported GPU generations on every CUDA test: 本 PR 在 tests/manual/launch_scripts/test_py_launch_scripts.py 的 _HARDWARE_A_RECORDING_REPRESENTS 中注册新启动器并固定 MI355X，延续了 #2701 建立起的「测试显式声明硬件」体系；同方向的 #3184 也在为遗漏的 CUDA 测试补硬件声明。
- PR #3184 ci: declare hardware on the two CUDA tests that landed after #2701: 与 #2701 同属 CI 硬件声明体系的补充，体现仓库对测试硬件显式化的持续收紧；与本 PR 的硬件映射注册属同一维护方向。