# PR #6790 完整报告

verl-project/verl

[trainer] feat: A runnable separate async trainer

合并时间：2026-06-17 22:25

原文链接：http://prhub.com.cn/verl-project/verl/pull/6790

## 执行摘要

- 一句话：修复分离异步 trainer 并启用运行
- 推荐动作：值得关注：这是分离异步训练器从 stub 变为可运行的关键一步，但功能仍不完整，建议跟进后续 PR 了解完整方案。设计中采用 start_rank 传递副本起始编号的做法可作参考。

## 功能与动机

此前 PPOTrainerSeparateAsync 在 __init__ 中直接抛出 NotImplementedError，导致分离异步训练器无法使用。PR 旨在使其达到可运行状态，实现类似全异步的独立 rollout 流程（参考 PR body: "a runnable separate async similar to fully async"）。后续将引入切换和 offload 策略。

## 实现拆解

1. 移除 NotImplementedError 并调整构造逻辑：
   - 在 verl/trainer/ppo/v1/trainer_separate_async.py 的 __init__ 中删除 raise NotImplementedError 语句，并将 super().__init__(config) 提前到验证之前，确保基类初始化在配置修改前完成。
   - 同时将 self.config.algorithm.rollout_correction.bypass_mode = True 移到 super().__init__ 之后，保证配置覆盖生效。
2. 解决混合 / 独立副本的 Ray 命名冲突：
   - 在 verl/workers/rollout/llm_server.py 的 LLMServerManager.__init__ 中新增 start_rank 参数（默认为 0），并存入 self.start_rank。
   - 修改 _initialize_llm_servers 方法：参数类型从 int = 0 改为 int = None，当为 None 时回退到 self.start_rank，使得独立副本可以从混合副本的末尾开始编号，避免 Ray 命名冲突。
   - 在 PPOTrainerSeparateAsync._setup 中，通过 len(self.llm_server_manager.rollout_replicas) 计算混合副本数，并作为 start_rank 传递给 LLMServerManager.create。
3. 移除冗余的回调方法：
   - 删除 on_validate_end 方法，因为验证后的切换逻辑已由 on_sample_end 覆盖，避免重复切换。
4. 添加 TODO 注释：

- 在 switch_to_rollout 和 switch_to_trainer 中增加 TODO，提示未来将关闭自动 offload 并根据切换策略进行 offload。

关键文件：

- verl/trainer/ppo/v1/trainer_separate_async.py（模块 训练器；类别 source；类型 core-logic；符号 PPOTrainerSeparateAsync, init, _setup, on_validate_end）：核心变更文件：移除 NotImplementedError 使类可运行，调整 __init__调用顺序，移除 on_validate_end，在 _setup 中传递 start_rank 给 LLMServerManager。
- verl/workers/rollout/llm_server.py（模块 worker；类别 source；类型 core-logic；符号 LLMServerManager, init, _initialize_llm_servers）：辅助变更文件：在 LLMServerManager 中新增 start_rank 参数，修改 _initialize_llm_servers 以支持从 self.start_rank 获取默认值，避免混合与独立副本的命名冲突。

关键符号：PPOTrainerSeparateAsync.init, PPOTrainerSeparateAsync._setup, LLMServerManager.init, LLMServerManager._initialize_llm_servers

## 关键源码片段

### verl/trainer/ppo/v1/trainer_separate_async.py

核心变更文件：移除 NotImplementedError 使类可运行，调整 __init__调用顺序，移除 on_validate_end，在 _setup 中传递 start_rank 给 LLMServerManager。

```
# verl/trainer/ppo/v1/trainer_separate_async.py

class PPOTrainerSeparateAsync(PPOTrainer):
    """Asynchronous PPO trainer
    1. Trainer and rollout are separate, trainer may switch to rollout if idle.
    2. Partial rollout is enabled.
    """

    def __init__(self, config: DictConfig):
        # 不再抛出 NotImplementedError，使类可运行
        train_batch_size = config.data.train_batch_size
        ppo_mini_batch_size = config.actor_rollout_ref.actor.ppo_mini_batch_size
        assert train_batch_size == ppo_mini_batch_size, (
            f"train_batch_size must be equal to ppo_mini_batch_size in separate async training, "
            f"but got {train_batch_size} and {ppo_mini_batch_size}"
        )
        assert config.actor_rollout_ref.rollout.nnodes > 0, "nnodes must be > 0 in separate async
        training"
        assert config.actor_rollout_ref.rollout.n_gpus_per_node > 0, (
            "n_gpus_per_node must be > 0 in separate async training"
        )
        assert config.actor_rollout_ref.rollout.checkpoint_engine.backend != "naive", (
            "please use nccl/nixl/mooncake, etc. backend for separate async training"
        )

        # 先调用基类初始化，再修改配置
```

```
        super().__init__(config)

        # TODO: Support Decoupled PPO: https://arxiv.org/abs/2505.24298
        self.config.algorithm.rollout_correction.bypass_mode = True

    def _setup(self):
        super()._setup()

        # 计算混合副本数，用作独立副本的起始 rank，避免 Ray 命名冲突
        hybrid_num_replicas = len(self.llm_server_manager.rollout_replicas)
        self.standalone_server_manager: LLMServerManager = LLMServerManager.create(
            config=self.config, start_rank=hybrid_num_replicas
        )
        # ... 其余初始化逻辑
```

## verl/workers/rollout/llm_server.py

辅助变更文件：在 LLMServerManager 中新增 start_rank 参数，修改 _initialize_llm_servers 以支持从 self.start_rank 获取默认值，避免混合与独立副本的命名冲突。

```
# verl/workers/rollout/llm_server.py

class LLMServerManager:
    """管理 LLM 服务器副本的启动与负载均衡"""

    def __init__(
        self,
        config: DictConfig,
        worker_group: RayWorkerGroup = None,
        rollout_resource_pool: RayResourcePool = None,
        start_rank: int = 0, # 新增参数：副本起始编号，用于避免 Ray actor 命名冲突
    ):
        self.config = config
        self.rollout_config = config.actor_rollout_ref.rollout
        self.model_config = config.actor_rollout_ref.model
        self.worker_group = worker_group
        self.rollout_resource_pool = rollout_resource_pool
        self.start_rank = start_rank # 存储起始 rank
        # ... 其余初始化

    async def _initialize_llm_servers(self, start_rank: int = None):
        """初始化 LLM 服务器副本

        Args:
            start_rank: 第一个副本的 rank。默认为 self.start_rank，
                        使得独立副本可以从混合副本的末尾编号，
                        避免 Ray 命名冲突。
        """
        if start_rank is None:
```

```
        start_rank = self.start_rank # 使用构造时传入的值
    # ... 创建副本逻辑，使用 start_rank 作为起始编号
```

## 评论区精华

仅有 gemini-code-assist[bot] 的一条评论，指出 _initialize_llm_servers 中 start_rank 的类型标注为 int 但默认值为 None，建议改为 Optional[int] 以通过类型检查。该评论未被解决，但 PR 已合并。

- start_rank 类型标注应改为 Optional[int] (style): 未被采纳，但类型不匹配为低风险，已合并。

## 风险与影响

- 风险：
  - 类型安全性：未采纳的类型标注建议可能导致 mypy 等类型检查器误报，但运行时无影响。
  - 回归风险：移除了 on_validate_end 方法，若未来有逻辑依赖该方法边界则可能失效，但目前基类或其他子类未使用。
  - 配置兼容性：LLMServerManager 新增了 start_rank 可选参数，现有调用（如混合模式）使用默认值 0，行为不变；但若外部代码直接实例化 LLMServerManager 并依赖位置参数，可能因新增参数而中断（Python 位置参数兼容）。
- 影响：
  - 对用户：分离异步训练器不再抛出异常，用户可通过 register_trainer("separate_async") 使用，但功能尚不完整（缺少切换与 offload 策略）。
  - 对系统：增大独立副本的 start_rank 避免与混合副本冲突，是架构上的必要修复。
  - 对团队：改动量小（仅 2 个文件，+17/-16），为后续开发奠定基础；需注意后续 TODO 的实现。
  - 风险标记：类型标注不匹配（低风险），功能尚不完整

## 关联脉络

- PR #6710 [trainer] feat: add unify trainer abstraction for sync and async training: 统一同步 / 异步 PPO trainer 抽象层，为本 PR 使用的基类和方法提供了支持。
- PR #6716 [trainer] fix: use FullyAsyncLLMServerClient for async trainer: 将 FullyAsyncLLMServerClient 迁移至核心模块，本 PR 依赖该客户端类。