verl-project/verl 2026 年第 24 周周报（06-08 ~ 06-14）

verl-project/verl

周期：2026-06-08 至 2026-06-14

来源 PR：33・重点 PR：24・自动生成

原文链接：http://prhub.com.cn/verl-project/verl/reports/2026-06-08-to-2026-06-14

1. 执行摘要

本周（06-08 至 06-14）verl 仓库共合并 33 个 PR，其中 24 个为重点 PR，平均重要性 6.02，平均洞察力 4.29。整体呈出现"架构重构 + 多硬件扩展 + 关键 bug 修复"三条主线。最重要的变化包括 PPO 训练器统一抽象（#6710）、Gemma4 多模态模型支持（#6715）、AMD ROCm 平台后端（#6702）以及长上下文 OOM 优化（#6593）。bugfix 仍占主导（19 个），标签分布显示 npu、rollout、vllm、trainer 是热点模块。本周集中修复了多个可能导致训练中断或崩溃的严重 bug，反映出框架在快速迭代中的稳定性增强。同时，CI 和测试覆盖显著增加，新增了 AMD ROCm、Qwen3.5 Ascend 等硬件 CI 工作流。

2. 本周重点变化

架构演进：PPO 训练器重构。PR #6710 引入了 V1 训练器基类 PPOTrainer，并提供同步、协同异步和分离异步三种实现。该重构统一了 PPO 训练器抽象，为后续异步训练奠定了基础。配套的 PR #6716 将 FullyAsyncLLMServerClient 迁移至核心模块，供异步 trainer 使用。不过，#6716 存在 Liskov 替换原则违反问题，需关注。

多模型扩展：Gemma4 支持。PR #6715 使 verl 能够使用 Gemma4 多模态模型进行 GRPO/PPO 训练。通过能力检测而非模型名称判断，采用了禁用 Gemma4Processor 严格校验等 hack，但缺少集成测试，风险未充分论证。

硬件平台扩展：AMD ROCm。PR #6702 新增 PlatformROCm，继承 PlatformCUDA 并覆写少量方法，使 ROCm 成为独立平台。同时 PR #6668 为 AMD MI300 添加 *e2e* PPO CI 工作流。加上本周多个 Ascend NPU 相关 PR（#6637、#6674、#6701 等），verl 的多硬件支持明显加强。

性能优化：分块 gather-logsumexp。PR #6593 通过数学恒等式和分块计算，避免在蒸馏训练中实例化巨大的 [B,T,V] log_softmax 张量，解决了长上下文 OOM 问题。该优化通过配置开关控制，默认启用。

关键 bug 修复集。本周修复了多个可能导致训练失败的重大 bug：空 token 序列崩溃（#6675），MTP drafter 权重丢失（#6661），FSDP rollout 缓冲区不同步（#5801），LoRA 权重 IPC 视图崩溃（#6688），DAPO 奖励配置门控条件（#6709）等。这些修复覆盖 rollout、trainer、reward、FSDP 等核心模块，显著提升了框架稳定性。

3. 模块与主题趋势

根据标签和文件热度，本周最活跃的模块依次是 rollout（vllm_rollout 相关文件修改最多）、trainer（PPO 训练器重构和 FSDP 修复）、megatron（多个 MTP 和配置兼容性修复）。标签统计显示 bugfix 占 19 个，其次是 test（8 个）和 npu（8 个）。说明本周以稳定性和测试增强为主基调。

- rollout 模块：重点修复了 MTP 混合睡眠权重丢失、空 token 序列、LoRA 视图崩溃、ZMQ socket 冲突等问题。vllm_rollout/utils.py 被修改 4 次，是本周最热文件。
- trainer 模块：训练器架构重构是重磅变化，同时修复了 FSDP graph retention 泄漏、Megatron 优化器 offload 遗漏等。
- reward 模块：修复了 DAPO 配置门控条件和 NaiveRewardManager 超时机制。
- 硬件支持：新增 AMD ROCm 平台和多项 Ascend NPU CI 工作流，体现了对多硬件生态的重视。
- 性能优化：分块 gather-logsumexp 是唯一一个性能优化 PR，但非常重要。

4. 风险观察

本周合并的 PR 中存在若干需要持续关注的风险：

1. Liskov 替换原则违反：PR #6716 FullyAsyncLLMServerClient.generate 方法未转发 **kwargs，可能导致运行时错误，且 review 中提出的建议未修复。对于异步训练的稳定性可能存在隐患。
2. 超时机制线程局限性：PR #6673 使用 SIGALRM 实现 per-sample 超时，但该信号仅主线程有效，在多线程 reward 计算场景下可能失效。作者已提及可改用进程池方案，但未在本 PR 解决。
3. 测试覆盖不足：多个核心路径变更（#6710 训练器重构、#6699 FSDP graph retention 修复、#6661 MTP sleep）缺乏充分的端到端测试，依赖 CPU 单元测试可能无法覆盖真实分布式场景。
4. 配置兼容性风险：PR #6702 PlatformROCm 依赖于 CUDA 兼容性，如果 ROCm 与 CUDA 行为差异未被充分测试，可能引入隐蔽错误。PR #6626 MTP 配置传播修复暴露了多 provider 架构下的配置覆盖问题，类似问题可能在其他配置项上重现。

5. 重点 PR 速览

- #6710: 统一 PPO 训练器抽象，引入 PPOTrainer 基类和三种实现，是本周最重要的架构变更。
- #6715: 支持 Gemma4 多模态模型 RL 训练，扩展了 verl 的模型支持范围。
- #6702: 新增 AMD ROCm 平台后端，使 AMD GPU 成为一等平台。
- #6593: 分块 gather-logsumexp 优化，解决长上下文 OOM，配置可选。
- #6661: 修复 MTP 混合睡眠中 drafter 权重丢失，通过配置门控选择睡眠等级。
- #6699: 修复 FSDP 训练中 per-micro-batch graph retention 导致的内存泄漏，通过 detach model_output 解决。
- #6675: 修复空 token 序列导致的 pad 和指标崩溃，处理了边缘情况。
- #5801: 修复 FSDP rollout 缓冲区不同步，拆分参数和缓冲区更新。

- #6620: 修复 DP>1 时 vLLM ZMQ socket 冲突，通过组合本地 DP 和 TP rank 生成唯一标识。
- #6631: 支持多模态路径占位符，扩展 dataset 输入类型。

6. 后续建议

- 跟踪训练器重构的后续 PR：建议团队重点关注 trainer_base.py 在异步模式下的成熟度，及时修复 review 中发现的 bug（配置路径错误、函数签名不匹配等）。
- 补充集成测试：针对本周合并的核心路径变更（#6710、#6699、#6661），建议在 GPU CI 中添加对应场景的端到端测试，避免回归。
- 关注 Liskov 违反问题：推动 #6716 的修复，确保 FullyAsyncLLMServerClient.generate 方法签名与父类一致，并转发 **kwargs。
- 增强 AMD ROCm 测试：虽然新增了 CI 工作流，但依赖外部 runner 可用性，建议内部搭建硬件环境确保测试持续运行。
- 继续内存优化：分块 gather-logsumexp 是一个好的开始，后续可探索 _forward_skip_lm_head 等方案进一步减少显存占用。
- 文档同步：本周新增了多个硬件文档（Ascend 最佳实践、VeOmni 后端等），建议建立文档版本与代码版本的对应机制，避免过期。