

PR #6957 完整报告

verl-project/verl

[algo] fix: normalize critic value loss over the global mini-batch, not per micro-batch

合并时间: 2026-07-07 16:56

原文链接: <http://prhub.com.cn/verl-project/verl/pull/6957>

执行摘要

- 一句话: 修复 Critic 损失微批归一化导致的梯度膨胀
- 推荐动作: 推荐精读, 理解损失归一化一致性的关键设计决策。该 PR 清晰地演示了如何从 Actor 对齐 Critic, 并包含详尽测试。Review 中关于不变异 config 对象的建议也值得在类似场景中推广。

功能与动机

Critic 值损失默认按每个微批独立归一化, 导致梯度累积膨胀约 K 倍 (K 为微批数), 且随 `use_dynamic_bsz` 等配置变化, 等效于隐式缩放 Critic 学习率。Actor 损失已通过 `ppo_loss` 传递全局归一化信息避免此问题, Critic 却遗漏了。详见 PR#6957 body 中的重现数据。

实现拆解

1. 修改 `value_loss` 函数 (`verl/workers/utils/losses.py`): 从 `data` 字段中提取 `dp_size`、`batch_num_tokens`、`global_batch_size`, 并检测 `config.loss_scale_factor`; 当任何全局归一化参数激活时, 指标聚合方式设为 SUM, 否则保持 MEAN; 将这些参数传递给 `compute_value_loss`。
2. 修改 `compute_value_loss` 函数 (`verl/trainer/ppo/core_algos.py`): 新增可选参数 `dp_size`、`batch_num_tokens`、`global_batch_size`、`loss_scale_factor`, 并在调用 `agg_loss` 时转发它们, 使损失计算能使用全局 token 数而非本地微批 token 数进行归一化。
3. 扩展 `CriticConfig` (`verl/workers/config/critic.py`): 添加 `loss_scale_factor` 字段, 默认 None, 用于 seq-mean-token-sum-norm 模式。
4. 修改 YAML 配置: 在 `verl/trainer/config/critic/critic.yaml` 和四个生成配置中增加 `loss_scale_factor` 项, 继承自 Actor 的同名字段。
5. 新增 CPU 回归测试 (`tests/trainer/ppo/test_value_loss_normalization_on_cpu.py`): 用无模型的数值模拟验证微批不变性、可变长度、DP_SIZE 模拟 FSDP 规约、指标聚合路径、`loss_scale_factor` 转发等, 共 26 个测试通过。

关键文件:

- `verl/workers/utils/losses.py` (模块 损失函数; 类别 source; 类型 core-logic; 符号 `value_loss`): 核心修改入口, 前向传递全局归一化参数并调整指标聚合
- `verl/trainer/ppo/core_algos.py` (模块 核心算法; 类别 source; 类型 core-logic; 符号 `compute_value_loss`): `compute_value_loss` 新增全局归一化参数, 透传给 `agg_loss`

- verl/workers/config/critic.py (模块 配置定义; 类别 source; 类型 data-contract; 符号 CriticConfig) : CriticConfig 新增 loss_scale_factor 字段
- tests/trainer/ppo/test_value_loss_normalization_on_cpu.py (模块 测试; 类别 test; 类型 test-coverage; 符号 _make_value_batch, _global_kwargs, _accumulate, test_value_loss_microbatch_invariant_all_modes) : 新增完整的 CPU 回归测试, 验证微批不变性和各种模式
- verl/trainer/config/critic/critic.yaml (模块 配置模板; 类别 config; 类型 configuration) : critic YAML 模板增加 loss_scale_factor 配置项

关键符号: value_loss, compute_value_loss, CriticConfig

关键源码片段

verl/workers/utils/losses.py

核心修改入口, 前向传递全局归一化参数并调整指标聚合

```
def value_loss(config: CriticConfig, model_output, data: TensorDict, dp_group=None):
    '''value loss'''
    vpreds = no_padding_2_padding(model_output['values'], data)

    # 提取全局归一化参数 (从 data 中获取)
    dp_size = data['dp_size']
    batch_num_tokens = data['batch_num_tokens']
    global_batch_size = data['global_batch_size']

    # 当任何全局归一化参数激活时, 使用 SUM 聚合
    if (
        dp_size > 1
        or batch_num_tokens is not None
        or global_batch_size is not None
        or config.loss_scale_factor is not None
    ):
        metric_aggregation = AggregationType.SUM
    else:
        metric_aggregation = AggregationType.MEAN

    # 选择字段并转换为 padded tensor
    data = data.select('values', 'returns', 'response_mask').to_padded_tensor()
    values = data['values']
    returns = data['returns']
    response_mask = data['response_mask'].to(bool)

    vf_loss, vf_clipfrac = compute_value_loss(
        vpreds=vpreds,
        values=values,
        returns=returns,
        response_mask=response_mask,
        cliprange_value=config.cliprange_value,
```

```

    loss_agg_mode=config.loss_agg_mode,
    dp_size=dp_size,
    batch_num_tokens=batch_num_tokens,
    global_batch_size=global_batch_size,
    loss_scale_factor=config.loss_scale_factor,
)

metrics = {
    'critic/vf_loss': Metric(value=vf_loss, aggregation=metric_aggregation),
    'critic/vf_clipfrac': vf_clipfrac.detach().item(),
    'critic/vpred_mean': masked_mean(vpreds, response_mask).detach().item(),
}

return vf_loss, metrics

```

评论区精华

Review 中 [gemini-code-assist](#) 指出不应变异共享的 `config` 对象来传输动态批量数据，而应直接透传参数，避免副作用和竞态条件。作者采纳建议，改为直接传递四个参数，并移除了 `config.global_batch_info` 的写入及 `CriticConfig` 中不再使用的 `global_batch_info` 字段。这一改进使接口更加清晰安全。

- 避免共享 `config` 对象变异，直接传递参数 (design): 作者接受建议，改为直接传递 `dp_size`、`batch_num_tokens`、`global_batch_size`、`loss_scale_factor` 四个参数，并移除了 `config.global_batch_info` 的写入及 `CriticConfig` 中不再使用的 `global_batch_info` 字段。

风险与影响

- 风险：低风险。核心改动是添加参数透传，未改变损失计算的数学公式。多 GPU FSDP 规约的模拟测试采用算术模拟而非真实多卡环境，但核心数值行为已得到验证。若 `loss_scale_factor` 在 `seq-mean-token-sum-norm` 模式下未正确设置，仍可能产生不一致（与 Actor 相同的限制）。新增的 CPU 测试覆盖了所有损失聚合模式和边界情况。
- 影响：直接影响所有使用 Critic 的 PPO 训练 (`adv_estimator=gae`, `use_critic=true`)，修正后训练梯度与微批划分无关，训练更稳定。指标报告方式调整，`critic/vf_loss` 在全局归一化时显示全局小批均值。对不使用 Critic 的 GRPO 等无影响。配置新增 `loss_scale_factor` 字段，向后兼容（默认 None）。
- 风险标记：核心损失函数变更，梯度数值行为改变，配置新增字段，FSDP 模拟测试有限

关联脉络

- PR #4211 Engine refactor: 该重构引入了问题：ppo_loss 获得了 `global_batch_info`，而 `value_loss` 没有。
- PR #5767 Description of issue (legacy path): 描述了同一问题但 diff 只触及已移除的旧文件，结论是当前路径不存在该问题（但实际仍然存在）。
- PR #3863 Pre-refactor mini_batch_full_token_count: 通过已废弃的机制解决，目前已失效。