PR #2153 完整报告

THUDM/slime

bugfix

合并时间：2026-06-30 18:29

原文链接：http://prhub.com.cn/THUDM/slime/pull/2153

执行摘要

- 一句话：将 with_entropy_grad 判断下移到 get_log_probs_and_entropy
- 推荐动作：该 PR 为简单的清理性 bugfix，不需重点精读，但体现了代码复用和职责分离的良好实践。

功能与动机

修复 get_log_probs_and_entropy 在外部调用（如 policy_loss_function 内部调用时已显式传入 with_entropy_grad）与实际场景之间的不一致：之前 with_entropy_grad 仅在 policy_loss_function 的调用处硬编码传入，导致直接调用 get_log_probs_and_entropy 的其他路径（如测试或实验代码）无法利用此优化，可能额外计算 entropy 梯度。

实现拆解

1. 在 get_log_probs_and_entropy 函数内部新增 with_entropy_grad 局部变量，其值根据参数 with_entropy 和配置项 args.entropy_coef 动态计算（with_entropy and getattr(args, "entropy_coef", 0.0) != 0）。
2. 将 with_entropy_grad 传递给内部调用的 calculate_log_probs_and_entropy。
3. 移除 policy_loss_function 中先前硬编码的 with_entropy_grad=args.entropy_coef != 0 参数，因为该逻辑已下沉到 get_log_probs_and_entropy 中。 该改动提升了代码复用性和调用安全性，确保所有对 get_log_probs_and_entropy 的调用都能自动获得正确的梯度控制。

关键文件：

- slime/backends/megatron_utils/loss.py（模块 损失计算；类别 source；类型 core-logic；符号 get_log_probs_and_entropy, policy_loss_function）：核心损失函数文件，集中了 logprob/entropy 计算和 PPO 损失逻辑；本 PR 在此文件移动了 with_entropy_grad 判定逻辑。

关键符号：get_log_probs_and_entropy, policy_loss_function

关键源码片段

slime/backends/megatron_utils/loss.py

核心损失函数文件，集中了 logprob/entropy 计算和 PPO 损失逻辑；本 PR 在此文件移动了 with_entropy_grad 判定逻辑。

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

def get_log_probs_and_entropy(logits, args, unconcat_tokens, total_lengths, response_lengths,
                              with_entropy=False, non_loss_data=True, top_p_token_ids=None,
                              top_p_token_offsets=None):
    ...
    tp_group = mpu.get_tensor_model_parallel_group()
    chunk_size = args.log_probs_chunk_size

    # 新增强制且统一的判定逻辑：只在 entropy_coef 非零时保留 entropy 梯度
    with_entropy_grad = with_entropy and getattr(args, "entropy_coef", 0.0) != 0

    full_tokens = _build_shifted_tokens(...)
    ...
    log_prob_full, entropy_full = calculate_log_probs_and_entropy(
        logits,
        full_tokens,
        tp_group,
        with_entropy=with_entropy,
        with_entropy_grad=with_entropy_grad, # 使用新计算的值
        chunk_size=chunk_size,
        log_prob_keep_mask=top_p_keep_mask,
    )
    ...

def policy_loss_function(args, batch, logits, sum_of_sample_mean):
    ...
    _, log_probs_and_entropy = get_log_probs_and_entropy(
        logits,
        args=args,
        unconcat_tokens=batch["unconcat_tokens"],
        total_lengths=total_lengths,
        response_lengths=response_lengths,
        with_entropy=True,
        # 删除了之前的 with_entropy_grad 参数，现在由内部自动计算
    )
    ...
```

评论区精华

该 PR 无 review 讨论或评论，变更简单明确，未出现争议。

- 暂无高价值评论线程

风险与影响

- 风险：风险极低。改动仅在 get_log_probs_and_entropy 内部添加了一行变量赋值，并移除了调用处的重复参数，语义等价。但需注意：若外部代码之前依赖 with_entropy_grad 参数被传入，此变更后该参数不再在调用处显式出现，但效果相同。建议检查其他调用点是否受影响（本 PR 未发现其他调用点）。
- 影响：影响范围小，仅限 PPO 训练中 entropy 梯度计算的优化逻辑，可小幅减少因冗余梯度计算导致的内存占用和计算开销。用户无感知，训练结果一致。
- 风险标记：缺少测试覆盖

关联脉络

- PR #2144 perf: fuse PPO logprob entropy computation: 本 PR 是对 #2144 引入的 with_entropy_grad 逻辑进行重构优化，将判定从调用点移至函数内部，提升一致性。