PR #2144 完整报告

THUDM/slime

perf: fuse PPO logprob entropy computation

合并时间：2026-06-29 16:08

原文链接：http://prhub.com.cn/THUDM/slime/pull/2144

执行摘要

- 一句话：融合 PPO logprob/entropy 计算，减少 allreduce 一半
- 推荐动作：建议团队成员精读 _VocabParallelLogProbEntropy 的 autograd 实现，作为融合通信的工程模式；同时也值得检查 CI 中新增的两个测试是否运行正常，确保数值稳定性。对于需要优化 PPO 训练性能的场景，此 PR 提供了解耦可复用的参考。

功能与动机

PR body 指出原本 logprob 和 entropy 分离计算存在冗余，通过 fused autograd helper 减少开销。移除不再使用的旧生产辅助函数。

实现拆解

1. 在 slime/utils/ppo_utils.py 中新增辅助函数 _maybe_all_reduce 和 _get_vocab_parallel_rank_size，用于在任意进程组上进行 allreduce 和获取 rank/size。
2. 定义 fused autograd 类 _VocabParallelLogProbEntropy，其 forward 方法中先通过 vocab_parallel_softmax 一次计算 softmax 及 sum_exp_logits，然后根据 with_entropy 和 log_prob_keep_mask 条件分别计算 logprob 和 entropy，共享 softmax 结果；backward 方法中根据链式法则为 logprob 和 entropy 的梯度分别计算并相加，传回 grad_input。
3. 在 slime/backends/megatron_utils/loss.py 的 get_log_probs_and_entropy 函数中调用新的 fused 函数 calculate_log_probs_and_entropy（实际为 _VocabParallelLogProbEntropy.apply），替代原来的分离调用，调整接口适配。
4. 移除 ppo_utils.py 中旧的 compute_log_probs 函数和 _VocabParallelEntropy 类，因为它们已被新实现取代。
5. 新增两个测试文件：tests/test_ppo_logprob_entropy.py 提供 CPU 环境下的数值正确性验证，使用 Gloo 分布式模拟并对比 unfused 参考实现；tests/test_ppo_logprob_entropy_gpu.py 提供 GPU 环境下的 Megatron 多卡校验，覆盖各种 mask 和 entropy 组合，确保 forward 和 backward 与旧版一致。
6. 在 CI 配置 .github/workflows/pr-test.yml.j2 和生成的 pr-test.yml 中将这两个新测试注册到对应 job 中（CPU 测试在无 GPU 的 job，GPU 测试在 2 GPU 的 job）。

关键文件：

- slime/utils/ppo_utils.py（模块 PPO 计算；类别 source；类型 core-logic；符号 _maybe_all_reduce, _get_vocab_parallel_rank_size, _VocabParallelLogProbEntropy, forward）：核心修改：新增 fused autograd 类 _VocabParallelLogProbEntropy，替换旧分离实现，并添加辅助函数。
- tests/test_ppo_logprob_entropy.py（模块 测试·CPU；类别 test；类型 test-coverage；符号 _free_port, _unfused_reference_logprob_entropy, _sum_in_partition_order, _reference_log_probs_with_partition_order）：CPU 测试覆盖，使用 Gloo 模拟分布式环境，对比 fused 与 unfused 参考实现。
- tests/test_ppo_logprob_entropy_gpu.py（模块 测试·GPU；类别 test；类型 test-coverage；符号 _free_port, _full_logits, _keep_mask, _weighted_loss）：GPU 测试覆盖，使用 Megatron 多卡验证 fused 版本与旧版数值一致性，包含各种 mask 与 entropy 组合。
- slime/backends/megatron_utils/loss.py（模块 损失函数；类别 source；类型 core-logic）：调用端升级：get_log_probs_and_entropy 改为使用 fused 接口。
- .github/workflows/pr-test.yml（模块 CI 配置；类别 infra；类型 infrastructure）：CI 配置中新增测试项目。
- .github/workflows/pr-test.yml.j2（模块 CI 模板；类别 infra；类型 infrastructure）：J2 模板注册新测试并调整格式。

关键符号：_VocabParallelLogProbEntropy.forward, _VocabParallelLogProbEntropy.backward, _maybe_all_reduce, _get_vocab_parallel_rank_size

关键源码片段

slime/utils/ppo_utils.py

核心修改：新增 fused autograd 类 _VocabParallelLogProbEntropy，替换旧分离实现，并添加辅助函数。

```
# slime/utils/ppo_utils.py（关键片段）
# （前略）

def _maybe_all_reduce(tensor, op, process_group):
    '''Conditional all-reduce to handle both distributed and non-distributed cases.'''
    if dist.is_available() and dist.is_initialized():
        dist.all_reduce(tensor, op=op, group=process_group)

class _VocabParallelLogProbEntropy(torch.autograd.Function):
    '''
    Fused autograd function that computes log-probability and optional entropy
    in one forward pass, sharing the softmax computation and its all-reduce
    cross communication. The backward similarly merges gradients from both losses.
    '''
    @staticmethod
    def forward(
        ctx,
```

```
    vocab_parallel_logits: torch.Tensor, # [seq_len, local_vocab]
    target: torch.Tensor, # [seq_len]
    log_prob_keep_mask: torch.Tensor | None,
    process_group,
    with_entropy: bool,
) -> tuple[torch.Tensor, torch.Tensor]:
    vocab_parallel_logits = vocab_parallel_logits.float()
    seq_len, local_vocab = vocab_parallel_logits.shape
    rank, _ = _get_vocab_parallel_rank_size(process_group)
    vocab_start = rank * local_vocab

    # ---- Shared softmax primitive ----
    def vocab_parallel_softmax(logits):
        # stable softmax with cross-shard max/sum reduces
        max_vals = logits.max(dim=-1, keepdim=True).values
        _maybe_all_reduce(max_vals, dist.ReduceOp.MAX, process_group)
        normalized = logits - max_vals
        exp = normalized.exp()
        sum_exp = exp.sum(dim=-1, keepdim=True)
        _maybe_all_reduce(sum_exp, dist.ReduceOp.SUM, process_group)
        softmax = exp / sum_exp
        return normalized, sum_exp, softmax, max_vals

    # ---- Log-probability branch (always computed) ----
    if log_prob_keep_mask is None:
        norm, sum_exp, softmax, max_v = vocab_parallel_softmax(vocab_parallel_logits)
        log_probs = ... # gather and subtract log(sum_exp); handle partition boundaries
    else:
        masked_logits = vocab_parallel_logits.masked_fill(~log_prob_keep_mask, float('-inf'))
        norm, sum_exp, softmax, max_v = vocab_parallel_softmax(masked_logits)
        log_probs = ... # similar calculation

    # ---- Entropy branch (conditional) ----
    entropy = torch.zeros(seq_len, ...)
    if with_entropy:
        if log_prob_keep_mask is None:
            # reuse softmax from logprob path
            sum_s_t = (softmax * vocab_parallel_logits).sum(dim=-1, keepdim=True)
            _maybe_all_reduce(sum_s_t, dist.ReduceOp.SUM, process_group)
            entropy = (max_v + sum_exp.log() - sum_s_t).squeeze(-1)
        else:
            # separate softmax on unmasked logits for entropy
            _, entropy_sum_exp, entropy_softmax, entropy_max = vocab_parallel_softmax(vocab_
            parallel_logits)
            sum_s_t = (entropy_softmax * vocab_parallel_logits).sum(dim=-1, keepdim=True)
            _maybe_all_reduce(sum_s_t, dist.ReduceOp.SUM, process_group)
            entropy = (entropy_max + entropy_sum_exp.log() - sum_s_t).squeeze(-1)

    return log_probs, entropy
```

评论区精华

无 reviewer 评论，作者自行合并。可能由于 PR 明确且已被充分验证。

- 暂无高价值评论线程

风险与影响

- 风险：主要风险来自数值精度：融合计算改变了 softmax 和梯度累加的顺序，可能导致浮点误差。但测试使用 atol=1e-7/1e-8 严格验证了前向和反向的数值一致性。第二个风险是移除了旧函数，可能被外部模块引用；但重构中已确认所有调用点已更新（只有 loss.py 一处）。第三个风险是 fused 实现使用了 _maybe_all_reduce，在非分布式环境下可能未初始化的 process group，但辅助函数已处理回退。总体风险可控。
- 影响：用户体验：PPO 训练性能提升，显存和通信带宽使用减少，梯度计算时间下降（具体幅度取决于模型规模）。系统层面：allreduce 调用次数减少一半，有助于减轻多节点通信压力。团队维护：代码结构简化，移除遗留函数，测试覆盖增强确保重构安全。影响范围限定在 PPO 相关训练的 Megatron backend，不影响其他算法（如 GRPO、Reinforce++）或后端（如 SGLang）。
- 风险标记：数值精度变化风险，旧函数移除破坏外部引用，多 GPU 通信顺序改变

关联脉络

- PR #2102 Support top_p mask: 此前 PR 引入了 top_p 掩码机制，影响了 logprob 计算路径；本 PR 将其融合进新的 autograd 实现，并修改了相同文件 loss.py。