

# PR #1284 完整报告

radixark/miles

Nemotron RL support

合并时间: 2026-08-05 13:49

原文链接: <http://prhub.com.cn/radixark/miles/pull/1284>

## 执行摘要

- 一句话: 新增 Nemotron-3-Ultra 混合架构 RL 训练支持
- 推荐动作: 值得精读。重点关注三处设计决策: (1) MTP labels 预移位的通用性——它修正了 Megatron-core MTP 损失的 off-by-one 语义, 是训练正确性修复的范例; (2) 对上游 PR 的显式依赖——shim 直接采用 Megatron-LM PR#51 的新签名, 展示了跨仓库协同改动的依赖管理方式, 同时也提示了版本对齐风险; (3) 工程减法——离线转换工具与 ipc\_collect 均经 review 论证后删除, 最终改动面 (+476/-4) 远小于过程版本, 体现了对最小变更的坚持。

## 功能与动机

Miles 需要支持 NVIDIA Nemotron-3-Ultra-550B-A55B 的 RL 训练。该模型是混合 Mamba2 + Attention + latent-MoE 架构 (108 层、512 experts top-22、moe\_latent\_size=2048), 现有社区 mbridge 后端 (tools/convert\_hf\_to\_torch\_dist.py) 无法转换此类架构, 因此必须借助 NVIDIA megatron.bridge (--megatron-to-hf-mode bridge) 与 miles 侧 NemotronHBridge shim 打通加载链路。同时, MTP 训练在混合架构上暴露出通用训练循环的缺陷: MTP labels 未按 Megatron-core 约定预移位导致 loss 异常、MambaModel.forward 不接受 loss\_mask 与 MTP 参数等问题需要修复。

## 实现拆解

实现按 5 个步骤拆解:

1. 新增 Python 启动器 (scripts/run\_nemotron\_3\_ultra\_550b\_a55b.py): 采用与 run\_qwen3\_30b\_a3b.py、run\_glm5\_744b\_a40b.py 相同的 ScriptArgs + typer 惯例, 继承 U.ExecuteTrainConfig。ScriptArgs.\_\_post\_init\_\_ 根据 model\_name 中是否含 Nlayer 自动推导 megatron\_model\_type 与切片模式; 提供 full-train / prepare / train 三个子命令; \_execute\_train 统一拼装 ckpt/rollout/perf/grpo/optimizer/misc 参数串并调用 U.execute\_train。
2. 并行度与 SGLang 引擎设计 (\_parallelism / \_sglang\_args): Mamba n\_groups=8 限制 attention/mamba 的 tensor-parallel 上限为 8。完整模型 (128 GPU) 采用 TP8/PP4/EP32, SGLang 侧使用 32-GPU 引擎 + DP-attention (dp=4), 使 attn\_tp = 32/4 = 8; 4 层切片 (单节点) 采用 TP1/PP1/EP8, SGLang 8-GPU 引擎 + dp=2。脚本明确不启用 rollout routing-replay (DP-attention 下 per-layer top-22 的 routing capturer 形状需修复), 文档说明 train/rollout logprob diff 约 0.01。

3. MTP 训练修复 (miles/backends/megatron\_utils/model.py) : forward\_step 中将 mtp\_labels 预移位为 torch.roll(batch["tokens"], shifts=-1, dims=-1), 使 MTP depth-k 目标对齐预训练 MTP head 期望的 token[i+k+1] (原始 tokens 会 off-by-one 导致 mtp\_loss 爆炸) ; train() 中初始化 mtp\_losses = None 并在 tracker 为空时跳过指标记录, 避免引用未绑定变量。
4. bridge shim 修复 (miles\_plugins/megatron\_bridge/nemotron\_h.py) : 禁用 MTP head 时同步清除 provider.mtp\_hybrid\_override\_pattern, 避免 finalize() 引用不存在的 Symbols.MTP\_SEPARATOR 报错; \_install\_mamba\_model\_loss\_mask\_shim 扩展 forward 签名, 将 miles 侧 GPT 风格的 mtp\_kwargs["mtp\_labels"] 透传给 MambaModel.forward 的 mtp\_labels 参数——该签名依赖上游 Megatron-LM PR#51 的 MambaModel.forward 增强。
5. 4 层切片配置与 CI 测试 (scripts/models/\*.sh + tests/e2e/.../test\_nemotron\_3\_ultra\_4layer\_ci.py) : 切片保留源层 0,1,7,8 (重编号 0..3), 块模式为 mamba/moe/attention/moe (ME\*E), 覆盖完整模型全部块类型, 512-experts / top-22 / moe\_latent\_size=2048 均不变; CI 测试注册 stage-c-8-gpu-h200 (900 秒), 开启 --ci-test 触发 Megatron -> SGLang 权重相等性检查, 并注册 grad\_norm / ppo\_kl / train\_rollout\_logprob\_abs\_diff 等指标门控。期间曾先行编写离线转换工具 tools/convert\_hf\_to\_torch\_dist\_bridge.py 并加入 CUDA-IPC ipc\_collect 修复, 后在 review 与对照实验后全部移除, 改为 bridge 直接加载 HF。

关键文件:

- scripts/run\_nemotron\_3\_ultra\_550b\_a55b.py (模块 训练脚本; 类别 source; 类型 entrypoint; 符号 ScriptArgs, post\_init, \_is\_pruned, \_hf\_checkpoint) : 全新训练启动器, 集中承载 Nemotron-H 的并行度设计 (Mamba n\_groups=8 约束下的 TP/EP/DP-attention 组合)、SGLang 引擎配置、GRPO 参数与 HF 直接加载路径, 是本次模型支持的主入口。
- miles/backends/megatron\_utils/model.py (模块 训练核心; 类别 source; 类型 core-logic; 符号 forward\_step, train) : 修改训练核心循环: MTP labels 预移位修复所有启用 MTP 训练的模型的 off-by-one 问题, 并门控 mtp\_losses 指标, 属于全局正确性修复。
- miles\_plugins/megatron\_bridge/nemotron\_h.py (模块 桥接层; 类别 source; 类型 core-logic; 符号 forward, \_build\_bridge\_subclass, provider\_bridge) : Nemotron-H bridge shim 的 MTP 相关修复: 禁用 MTP head 时清理 hybrid override pattern 避免 finalize 崩溃; 扩展 MambaModel.forward shim 透传 mtp\_labels, 是模型能跑通 MTP 训练的关键粘合层。
- tests/e2e/megatron/model\_scripts/test\_nemotron\_3\_ultra\_4layer\_ci.py (模块 CI 测试; 类别 test; 类型 test-coverage; 符号 \_args, prepare, execute) : 新增 8-GPU H200 冒烟测试, 验证 4 层切片训练脚本可运行且 Megatron -> SGLang 权重同步相等性, 是模型支持的 CI 保障。
- scripts/models/nemotron-3-ultra-550b-a55b-4layer.sh (模块 模型配置; 类别 config; 类型 configuration) : 4 层切片模型配置, 说明层选择策略 (ME\*E 覆盖全块类型) 与全量模型一致的 MoE 超参, 保证权重转换路径相同。

- scripts/models/nemotron-3-ultra-550b-a55b.sh (模块 模型配置; 类别 config; 类型 configuration) : 完整 108 层模型的 MODEL\_ARGS 配置, 支撑 16 节点 128 GPU 训练, 是脚本命令组装的数据契约。

关键符号: ScriptArgs.post\_init, \_is\_pruned, \_parallelism, \_sglang\_args, \_execute\_train, forward\_step, train, provider\_bridge, \_install\_mamba\_model\_loss\_mask\_shim

## 关键源码片段

### scripts/run\_nemotron\_3\_ultra\_550b\_a55b.py

全新训练启动器, 集中承载 Nemotron-H 的并行度设计 (Mamba n\_groups=8 约束下的 TP/EP/DP-attention 组合)、SGLang 引擎配置、GRPO 参数与 HF 直接加载路径, 是本次模型支持的主入口。

```
def _parallelism(args: ScriptArgs) -> tuple[int, int, int, int]:
    """(tp, pp, ep, etp)。Mamba n_groups=8 限制了 attention/mamba 的 tp 上限为 8。"""
    total_gpus = args.num_nodes * args.num_gpus_per_node
    if _is_pruned(args):
        # 4 层切片只需一个节点: 把所有 rank 都分给 expert parallel
        # (512 experts / EP8 = 每 rank 64 个 expert), attention/mamba 保持 tp 1。
        return 1, 1, total_gpus, 1
    if total_gpus == 128:
        return 8, 4, 32, 1
    raise NotImplementedError(f"No parallelism config for {total_gpus} GPUs with {args.model_name}")

def _sglang_args(args: ScriptArgs) -> str:
    total_gpus = args.num_nodes * args.num_gpus_per_node
    if _is_pruned(args):
        gpu_per_engine, dp_size, mem_fraction = args.num_gpus_per_node, 2, 0.6
    else:
        # 550B (约 1.1TB BF16) 放不进单个 8-GPU 引擎, 改用 32-GPU 引擎 +
        # EP=32 + DP-attention (dp=4), 让 attention/mamba 以 attn_tp = 32/4 = 8 运行。
        gpu_per_engine, dp_size, mem_fraction = 32, 4, 0.7
    assert total_gpus % gpu_per_engine == 0, f"{total_gpus=} must be a multiple of {gpu_per_engine=}"
    attn_tp = gpu_per_engine // dp_size
    assert (
        gpu_per_engine % dp_size == 0 and 8 % attn_tp == 0
    ), f"attn_tp = {gpu_per_engine}/{dp_size} = {attn_tp} must divide Mamba n_groups=8"
    # DP-attention 是让 attention/mamba 并行度回落的关键开关, 同时需要 SGLang
    # 引擎共享权重, 因此脚本统一开启 --sglang-enable-dp-attention。
    return (
        f"--rollout-num-gpus-per-engine {gpu_per_engine} "
        f"--sglang-ep-size {gpu_per_engine} "
        f"--sglang-dp-size {dp_size} "
        "--sglang-enable-dp-attention "
        f"--sglang-mem-fraction-static {mem_fraction} "
```

)

## miles/backends/megatron\_utils/model.py

修改训练核心循环：MTP labels 预移位修复所有启用 MTP 训练的模型的 off-by-one 问题，并门控 mtp\_losses 指标，属于全局正确性修复。

```
if args.enable_mtp_training:
    # Megatron-core 的 MTP loss 期望 next-token 移位的 labels
    # (labels[i] == token[i+1])，每个 MTP depth 会再 roll -1 一次。
    # 原始 tokens 会让 MTP target off-by-one，使 mtp_loss 相对预训练
    # MTP head 直接爆炸，所以这里先预移位一次。
    forward_kwargs["mtp_kwargs"] = {"mtp_labels": torch.roll(batch["tokens"], shifts=-1, dims=-1)}

# ... 训练循环内部 (train) 围绕 tracker 的防御性取值：
# MTPLossLoggingHelper.tracker 在部分模型 / 分支下可能为空，
# 若直接引用 mtp_losses 会在日志阶段抛 UnboundLocalError，
# 因此显式初始化 None 并在指标组装时门控。
mtp_losses = None
if "values" in tracker:
    values = tracker["values"]
    # 先做 group 归约再取标量，保证各 rank 日志一致。
    if (x := tracker.get("reduce_group")) is not None:
        torch.distributed.all_reduce(values, group=x)
    if (x := tracker.get("avg_group")) is not None:
        torch.distributed.all_reduce(values, group=x, op=torch.distributed.ReduceOp.AVG)
    mtp_losses = (tracker["values"] * mtp_loss_scale).item()
    MTPLossLoggingHelper.clean_loss_in_tracker()
```

## miles\_plugins/megatron\_bridge/nemotron\_h.py

Nemotron-H bridge shim 的 MTP 相关修复：禁用 MTP head 时清理 hybrid override pattern 避免 finalize 崩溃；扩展 MambaModel.forward shim 透传 mtp\_labels，是模型能跑通 MTP 训练的关键粘合层。

```
def _install_mamba_model_loss_mask_shim() -> None:
    """让 MambaModel.forward 接受 loss_mask 与 mtp_labels（依赖上游 megatron PR#51 的签名）。"""
    from megatron.core.models.mamba import MambaModel

    if getattr(MambaModel, "_miles_loss_mask_shim_installed", False):
        return

    _orig_forward = MambaModel.forward

    def forward(self, *args, loss_mask=None, mtp_kwargs=None, **kwargs):
        # miles 的通用训练循环按 GPTModel 风格传 mtp_kwargs，而 MambaModel 需要
        # 扁平化的 mtp_labels 参数。这里做适配：MTP loss 以 mtp_labels 为门控，
        # 主路径保持 labels=None 仅返回 logits 供 policy-gradient loss 使用。
        # 注意：loss_mask 直接透传依赖上游 Megatron-LM PR#51 为该函数新增参数，
        # 若上游未合入，此 shim 会因 unexpected keyword 抛 TypeError。
```

```
mtp_labels = (mtp_kwargs or {}).get("mtp_labels")
return _orig_forward(self, *args, loss_mask=loss_mask, mtp_labels=mtp_labels, **kwargs)
```

```
MambaModel.forward = forward
```

```
MambaModel._miles_loss_mask_shim_installed = True
```

## 评论区精华

核心讨论集中在三点：

1. MambaModel.forward 的 loss\_mask 兼容性 (gemini-code-assist[bot] 高优先级)：机器人指出 "The original MambaModel.forward (captured in \_orig\_forward) does not accept loss\_mask ... Passing loss\_mask=loss\_mask to \_orig\_forward will raise a TypeError", 建议仅透传 mtp\_labels。最终合并代码仍保留 loss\_mask=loss\_mask 透传，说明团队有意依赖上游 Megatron-LM PR#51 为 MambaModel.forward 新增该参数——这是一个显式的跨仓库协同决策，而非疏漏。
  2. 离线转换工具存废 (yueming-yuan 质疑)："why do we need this script? given that we usually just load hf weight directly when we use megatron bridge", 随后 Yuming 自行删除该文件，Zhichenzzz 回复 "should remove it"。提交 e978259 说明理由：bridge 模式已能直接从 HF 加载，离线转换只是摊销映射成本，却引入 TP/PP/EP 匹配约束，直接加载更简洁。
  3. ipc\_collect 的必要性 (yueming-yuan 质疑)："is this needed? in my understanding ipc should be automatically collected in weight update (so other models work well)". Zhichenzzz 回应 "repeated ipc collect, will delete this and that one in sglang", 最终两处 ipc\_collect 全部删除，最后提交 (ce987c3) 注明 4 层 A/B 对照 (6 轮、offload 开、含 / 不含 --ci-test) 未复现内存增长，依赖下一轮 IPC 活动的自动 pending-free 回收。
- MambaModel.forward shim 透传 loss\_mask 的 TypeError 风险 (correctness): 合并后的 shim 仍保留 loss\_mask=loss\_mask 透传，团队显式依赖上游 Megatron-LM PR#51 为 MambaModel.forward 新增 loss\_mask 与 mtp\_labels 参数；离线转换工具随后被删除，shutil.move 问题随之消失。
  - 离线转换工具 convert\_hf\_to\_torch\_dist\_bridge.py 是否必要 (design): 删除该工具，训练统一走 --hf-checkpoint + --ref-load 指向 HF 目录的 bridge 直接加载路径，保持与实时训练同一转换链路。
  - colocate 权重同步后的 CUDA-IPC 手动回收是否必要 (performance): 两处 ipc\_collect 全部删除，依赖下一轮 IPC 活动的自动 pending-free 回收；actor.py 最终与 main 分支完全一致。

## 风险与影响

• 风险：

1. 核心训练循环修改 (model.py)：mtp\_labels 预移位影响所有启用 --enable-mtp-training 的模型（如 qwen3 系列）。若某个模型或数据流水线此前已做移位，将出现双重移位；按 Megatron-core 语义这是统一修复，但需要回归其他 MTP 训练用例。

2. 依赖外部上游 PR: MambaModel shim 的 forward 签名 (loss\_mask / mtp\_labels) 依赖 Megatron-LM PR#51 合入; 若上游未合入或签名不一致, Mamba 模型前向将直接 TypeError, 且该依赖未在 shim 中做版本检查或降级处理。
3. CI 依赖外部 checkpoint: 4 层测试依赖 CharyZeng 组织发布的切片 checkpoint, 提交历史上就出现过 checkpoint 未发布导致 hf download 404、CI 40 秒失败的先例 (11a1a7e 曾禁用测试)。checkpoint 一旦下线, CI 立即失败。
4. 脚本硬编码路径: /root/datasets、/root/models、/root/Megatron-LM 与 dapo-math-17k 数据集均为固定约定, 换环境需额外配置。
5. routing-replay 未启用: 完整模型 train/rollout logprob diff 约 0.01, 对需要严格对齐的 RL 场景可能存在精度影响, 属于已知限制。- 影响: 用户侧: 新增 Nemotron-3-Ultra-550B-A55B 的完整 RL 训练能力 (16 节点 128 GPU 全量 + 单节点 4 层切片), 并修复了通用 MTP 训练 labels 错位问题, 利好所有启用 MTP 的模型。系统侧: 训练循环 forward\_step 的 MTP 行为变化是全局性的, 需回归验证; SGLang 侧依赖 DP-attention 引擎配置与 dp-aware 路由, 跨模块耦合加深。团队侧: 新增模型脚本 + bridge 插件 shim + CI 测试的维护负担, 且形成对上游 Megatron-LM 与 sglang 两个仓库 PR 的版本依赖, 后续升级需同步对齐。CI 新增 stage-c-8-gpu-h200 套件 (约 900 秒), 增加流水线时长。- 风险标记: 核心训练循环修改, 依赖外部上游 PR, CI 依赖外部 checkpoint, 新增模型配置, 存在后续回归修复

## 关联脉络

- PR #2041 Address engines by base URL when the router is dp-aware: 本 PR 的 SGLang 引擎配置大量使用 DP-attention (--sglang-enable-dp-attention), 后续 PR 修复了 dp-aware 路由下 abort 无法触达引擎的问题, 构成对 DP-attention 依赖链路的补全。
- PR #2077 Reject --disable-weights-backuper for LoRA + colocate + offload-train: 本 PR 的 4 层 CI 测试恰好使用 --disable-weights-backuper 与 --ci-test 组合; 后续 PR 对该组合加了参数校验 (后又因过宽而回滚), 说明该 flag 组合存在隐含约束。
- PR #2203 Revert "Reject --disable-weights-backuper for LoRA + colocate + offload-train" (#2077): 回滚了 2077 的过宽断言, 与本 PR 测试使用的 flag 组合直接相关, 需关注最终约束形态。