PR #1759 完整报告

radixark/miles

(2/2) refactor(session): assemble training samples on the session server; records never leave it

合并时间：2026-07-30 08:12

原文链接：http://prhub.com.cn/radixark/miles/pull/1759

执行摘要

- 一句话：训练样本组装移入 session server，records 不再出服务器
- 推荐动作：值得精读。该 PR 是一个典型的「传输与计算位置重构」案例：用单一 wire 契约（SAMPLES_VALUE_SPEC）把服务端计算与客户端覆盖解耦，用 deepcopy + 默认值守卫保证输入模板不被污染，并以 golden 测试锁定行为。重点关注三个设计决策：① 字段白名单 + 固定 dtype + strict 校验如何防止静默精度损失；② 超时 fail-loud 与边界 ABORT 的权衡；③ 服务端事件循环同步组装的取舍。

功能与动机

PR body 明确指出：旧的 rollout driver 需要先下载每条 session record 再做计算、截断、合并，而每个 record 都包含完整轨迹前缀，导致传输量随轮数二次方增长，driver 还要序列化、传输、解析这些只为样本组装服务的数据。因此本 PR 确立 HTTP 作为完整的基线传输方式，让 records 永远不离开 session server；同时刻意不引入 Mooncake 依赖，为后续可选优化留出接口。

实现拆解

1. 新增 wire 编解码层：新增 miles/rollout/session/samples/codec.py，用 SAMPLES_VALUE_SPEC 定义 11 个可跨线字段的 codec 类型（tensor / tensor_list / json）、固定 dtype、strict 校验标记和 null 还原值；encode_samples 将服务端组装的 Sample 打包进单个 safetensors 容器（元数据以 JSON 塞入 _samples_meta 张量），decode_samples_and_merge_input_sample 在 driver 侧把计算字段覆盖到输入 Sample 的 deepcopy 上，并通过 assert_input_sample_defaults 守卫防止旧管线遗留字段污染合并结果。
2. 服务端组装入口：miles/rollout/session/core.py 新增 SessionCore.collect_samples，与 get_session 共享新建的 _session_metadata 辅助方法（保证两个接口的 metadata 永不漂移）；组装流程为 compute_samples_from_openai_records → truncate_samples_by_total_tokens（turn 级预算）→ merge_samples，空记录与全截断分别以 empty_reason 为 no_records / all_truncated 的 200 空响应返回，确定性组装失败（断言 / 数值错误）映射为 422 文本响应。
3. 路由与客户端切换：miles/rollout/session/sessions.py 注册 /sessions/{id}/samples 路由并保证在 catch-all proxy 之前匹配；miles/rollout/generate_utils/openai_endpoint_utils.py 用 collect_samples 替换 collect_records，内部通过新增的

miles/utils/http_utils.py::post_bytes_no_retry 做一次非重试二进制 POST，超时、传输失败或非 2xx 均向上抛错，finally 中尽力 DELETE session 且 DELETE 失败不掩盖 POST结果。

4. 驱动侧生成逻辑收敛：miles/rollout/generate_hub/agentic_tool_call.py 删除 driver 侧组装 / 截断 / 合并代码，改为直接消费 collect_samples 返回的 SamplesReply，叠加 agent 元数据后应用 session 元数据；超时被转换为 ABORTED 样本，非超时错误继续传播（提交 9afc62be 专门修复此语义）；miles/rollout/session/samples/merge.py 的 compute_samples_from_openai_records / _compute_sample_from_openai_record 不再接收 input_sample，直接产出空白 Sample，DRY 地让服务端与 driver 共享同一实现。
5. 测试与文档配套：新增 tests/fast/router/test_session_samples_op.py（golden 样本、截断、422、路由注册顺序）、tests/fast/rollout/session/test_samples_codec.py（wire round-trip、畸形 payload 拒绝、默认值守卫）、扩写 tests/fast/rollout/generate_utils/test_openai_endpoint_utils.py（单 POST 后 DELETE、非 2xx/ 超时 / 删除失败语义）与 tests/fast/rollout/generate_hub/test_multi_turn.py（超时 ABORT、其他错误传播）；docs/user-guide/rollout-endpoints.md 与 requirements.txt（safetensors 依赖）同步更新。

关键文件：

- miles/rollout/session/samples/codec.py（模块 样本编解码；类别 source；类型 core-logic；符号 ValueSpec, SamplesReply, _asarray_wire, encode_samples）：新增的 wire 编解码核心：SAMPLES_VALUE_SPEC 字段白名单、safetensors 打包、driver 侧覆盖合并与默认值守卫，是本次重构的契约中心。
- miles/rollout/session/core.py（模块 会话服务；类别 source；类型 core-logic；符号 _samples_response, get_session, _session_metadata, collect_samples）：服务端组装入口：新增 collect_samples（compute→truncate→merge），抽取 _session_metadata 保证与 get_session 的 metadata 一致。
- miles/rollout/generate_utils/openai_endpoint_utils.py（模块 端点客户端；类别 source；类型 core-logic；符号 collect_records, collect_samples）：客户端关键切换：collect_records 被 collect_samples 取代，一次非重试二进制 POST + finally 中尽力 DELETE，超时 /non-2xx fail-loud。
- miles/rollout/generate_hub/agentic_tool_call.py（模块 生成编排；类别 source；类型 core-logic；符号 generate, collect_samples, collect_timed_out）：驱动侧生成逻辑：删除本地组装，改为消费 SamplesReply；超时映射 ABORTED、其他错误传播，是超时语义修复的落点。
- miles/rollout/session/samples/merge.py（模块 样本组装；类别 source；类型 refactor；符号 compute_samples_from_openai_records, _compute_sample_from_openai_record）：组装函数签名重构：compute_samples_from_openai_records / _compute_sample_from_openai_record 不再接收 input_sample，改产出空白 Sample，成为服务端与 driver 共享的唯一实现。
- miles/utils/http_utils.py（模块 HTTP 工具；类别 source；类型 core-logic；符号 post_bytes_no_retry, _do）：新增 post_bytes_no_retry 基础 HTTP 原语：单次 POST、总超时、非 2xx 带响应体抛错，是客户端 fail-loud 语义的传输基础。

- tests/fast/router/test_session_samples_op.py（模块 样本接口测试；类别 test；类型 test-coverage；符号 _UnusedBackend, do_proxy, _build_core, core）：golden 测试：用真实 tokenizer 驱动 collect_samples，断言合并后 Sample 的每个字段值（含截断、R3、metadata 覆盖顺序），并验证 404/422/ 路由顺序。
- tests/fast/rollout/session/test_samples_codec.py（模块 编解码测试；类别 test；类型 test-coverage；符号 _computed_sample, _mutated_payload, TestSamplesWireCodec, test_round_trip_overlays_computed_and_keeps_template）：wire codec 契约测试：round-trip、多样本隔离、默认值守卫、畸形 payload 参数化拒绝，保证编解码层正确性。

关键符号：SessionCore.collect_samples, SessionCore._session_metadata, OpenAIEndpointTracer.collect_samples, post_bytes_no_retry, encode_samples, decode_samples_and_merge_input_sample, assert_input_sample_defaults, compute_samples_from_openai_records, agentic_tool_call.generate

关键源码片段

miles/rollout/session/core.py

服务端组装入口：新增 collect_samples（compute→truncate→merge），抽取 _session_metadata 保证与 get_session 的 metadata 一致。

```
import json
import logging
import time
from dataclasses import dataclass

from starlette.responses import Response

from miles.rollout.generate_utils.sample_utils import merge_samples
from miles.rollout.session.errors import (
    MessageValidationError,
    SessionNotFoundError,
    TokenizationError,
    UpstreamResponseError,
)
from miles.rollout.session.linear_trajectory import SessionRegistry
from miles.rollout.session.samples.codec import encode_samples
from miles.rollout.session.samples.merge import compute_samples_from_openai_records,
truncate_samples_by_total_tokens
from miles.rollout.session.types import GetSessionResponse, SessionRecord

logger = logging.getLogger(__name__)

JSON_MEDIA_TYPE = "application/json"


def _samples_response(payload: bytes) -> Response:
    """samples-op 的统一响应：一个 safetensors 二进制负载。"""
    return Response(content=payload, status_code=200, media_type="application/octet-stream")
```

```
class SessionCore:
    """HTTP session 操作，持有 SessionRegistry 与代理 backend。"""

    def _session_metadata(self, session_id: str, session) -> dict:
        """组装/检查共用的元数据字典：`get_session`（records 调试转储）与
        `collect_samples`（samples op）都用它，保证两者永不产生漂移。"""
        metadata: dict = {}
        try:
            mismatch = self.registry.compute_session_mismatch(session)
        except TokenizationError:
            logger.exception("Failed to compute tito_session_mismatch for session %s", session_id)
            mismatch = None
        if mismatch is not None:
            metadata["tito_session_mismatch"] = mismatch
        metadata["accumulated_token_ids"] = session.token_ids
        metadata["max_trim_tokens"] = self.registry.tito_tokenizer.max_trim_tokens
        return metadata

    async def collect_samples(self, session_id: str, *, max_seq_len: int | None) -> Response:
        """从本 session 的 records 组装训练样本。

        流程：compute（records → 逐 turn Sample）→ 按 max_seq_len 截断
        → 无条件 merge 成单个轨迹样本。确定性错误（断言/数值）返回 422；
        其它异常正常上抛。与 `get_session` 一样在事件循环上同步执行，
        依赖锁自由读取保证看到一致的 records 列表。
        """
        session = self.registry.get_session(session_id)
        metadata = self._session_metadata(session_id, session)
        tokenizer = self.registry.tokenizer
        if not session.records:
            return _samples_response(encode_samples([], metadata, empty_reason="no_records"))
        try:
            samples = compute_samples_from_openai_records(
                self.args,
                session.records,
                tokenizer,
                accumulated_token_ids=metadata.get("accumulated_token_ids"),
                max_trim_tokens=metadata.get("max_trim_tokens", 0),
            )
            if max_seq_len is not None:
                # 截断发生在 merge 之前：它是 turn 级预算决策，turn 结构只在合并前存在
                samples = truncate_samples_by_total_tokens(samples, max_seq_len, tokenizer)
            if not samples:
                return _samples_response(encode_samples([], metadata, empty_reason="all_
                truncated"))
            samples = [merge_samples(samples, tokenizer)]
        except (AssertionError, ValueError) as exc:
            # 确定性组装失败：把断言文本回给驱动端，而不是让服务器 500
            return Response(content=str(exc).encode(), status_code=422, media_type="text/plain")
```

```
    return _samples_response(encode_samples(samples, metadata))
```

评论区精华

Review 中最有价值的交锋是 Shi-Dong 对 OpenAIEndpointTracer.collect_samples 超时语义的质疑：

> "Something I don't understand: what if TimeoutError is raised here? Wouldn't the error be propagated out and crash the rollout group?"

guapisolo 先回复 "let me carefully check this"，随后在提交 9afc62be 中落地修复：agentic_tool_call.generate 捕获 asyncio.TimeoutError 并映射为单个 ABORTED 样本返回，而其他收集错误继续传播；与此配套的测试 TestAgentCollectionFailure.test_collect_timeout_aborts_sample_but_other_errors_propagate 锁定了该行为。另一个评论是 guapisolo 对 codec.py 的说明："This is a vibed file, but verified by e2e CIs. No tito mismatch rate regression."，表明该文件部分凭经验编写，但通过端到端 CI 与 TITO mismatch 率回归验证兜底。最终 Shi-Dong 给出 APPROVED（"LGTM!"）。

- collect_samples 超时是否会拖垮 rollout group (correctness): 在 agentic generate 边界捕获 asyncio.TimeoutError 并映射为 ABORTED 样本；非超时错误继续传播。新增测试 test_collect_timeout_aborts_sample_but_other_errors_propagate 锁定该语义。
- codec.py 被标记为 vibed file 的验证方式 (testing): 接受 e2e CI 与 TITO mismatch 率回归作为验证手段；配套 golden 单元测试覆盖主要路径。

风险与影响

- 风险：
  1. 核心训练数据链路变更：样本组装从 driver 移到 session server，任何组装逻辑缺陷会直接污染训练样本；golden 测试覆盖了两轮轨迹与截断场景，但 R3 replay 解码、TITO mismatch 等复杂路径依赖 e2e CI 兜底（codec.py 被作者自述为 vibed file）。
  2. 超时语义变化：旧 collect_records 静默吞掉超时返回空 records（等价 ABORT），新路径在客户端 fail-loud、在 generate 边界转 ABORTED；若未来新增调用方未按此约定处理，超时仍可能向上传播导致 rollout group 失败。
  3. wire 契约脆弱性：SAMPLES_VALUE_SPEC 是唯一字段 / 类型契约，若新增计算字段忘记加入白名单，将被静默丢弃；decode_samples_and_merge_input_sample 对畸形 payload 采用 fail-loud 策略（KeyError/ValueError 直接抛出），服务端与 driver 版本不一致时会直接中断样本收集。
  4. 服务端 CPU 占用：SessionCore.collect_samples 在事件循环上同步执行组装（锁自由读取），长轨迹的组装 / 截断 / 合并可能阻塞该 session 的其它请求；当前设计刻意与 get_session 保持一致，但高并发下存在事件循环阻塞风险。
  5. DELETE 竞态：POST 成功后 finally 中执行 DELETE，若 DELETE 在 POST 返回前到达，服务端可能因 session 已删除而 404；测试只覆盖了 DELETE 失败被容忍，未覆盖该竞态顺序。 - 影响：对用户 / 系统：训练样本组装彻底改道 HTTP，records 不再离开 session server，大幅降低 driver 到 server 的带宽与 driver 侧 CPU/ 内存占用，尤其利好长多轮 agentic 轨迹；新增 POST /sessions/{id}/samples 接口与 safetensors 二

进制响应成为新的传输基线，后续 Mooncake 等优化可无缝叠加。对团队：session 包职责扩展为「轨迹存储 + 样本组装 + wire 编解码」，samples/merge.py 成为服务端与 driver 共享的唯一组装实现；测试体系新增两组高价值 golden/ 契约测试，为后续演进提供回归保障。对兼容性：driver 与 session server 必须同版本部署，否则 wire 契约（字段、dtype）可能不匹配；requirements.txt 新增 safetensors 依赖。 - 风险标记：训练数据核心链路重构，新 wire 协议需双端同版本，服务端事件循环同步组装，超时语义变更，codec 依赖 e2e 兜底

关联脉络

- PR #2028 session: collect speculative-decoding counters: 同样修改 miles/rollout/session/samples/merge.py 与 tests/fast/rollout/session/test_samples.py，属于 session samples 管线的持续演进，且 #1759 的 merge.py 签名重构会直接影响 #2028 的计数器采集落点。
- PR #1965 dashboard: fix phase visibility for manager events and idle processes: 涉及 session 状态与事件可见性，与 #1759 同属 session 服务链路，但无直接代码交集，关联度较弱。