PR #7491 完整报告

verl-project/verl

[rollout] fix: preserve default AgentLoop extra fields

合并时间：2026-08-21 15:49

原文链接：http://prhub.com.cn/verl-project/verl/pull/7491

执行摘要

- 一句话：修复 AgentLoop 默认 extra_fields 序列化 KeyError
- 推荐动作：值得精读。这是一个「5 行修复 + 完整回归覆盖」的教科书式 bugfix：根因分析精确到 Pydantic exclude_unset 与默认可变字段的语义组合，修复用 setdefault 保持了「序列化只读」的不变量。值得借鉴的设计点： 1) 序列化 / 导出方法不应修改模型自身状态； 2) 对「默认可变字段」的序列化缺陷，用三个最小用例把行为钉死； 3) AI 辅助（Codex）参与实现时，PR body 完整披露并保留人工 review 的 draft 流程，适合作为团队 AI 协作规范的样例。

功能与动机

关联 Issue #7486 报告：extra_fields 是带默认值的可选字段，构造 AgentLoopOutput 时不传即合法，但 as_dict() 在 model_dump(exclude_unset=True) 后无条件索引 output["extra_fields"]，而 Pydantic 不会把默认字段加入 fields_set，构造后原地修改默认 dict也不会补记，于是合法实例直接抛KeyError:'extra_fields'，动态添加的字段也全部丢失。该缺陷可经官方代码触达：verl/trainer/ppo/v1/agent_loop_tq.py 对每条 TransferQueue 轨迹调用 as_dict()，langgraph_agent 官方 recipe 的 fallback 轨迹不显式传 extra_fields，触发后轨迹无法发布、prompt 被标记为 failed。

实现拆解

1. 根因定位（来自 Issue #7486 的分析）

- as_dict() 先执行 model_dump(exclude_unset=True)，随后无条件访问 output["extra_fields"].pop(...)。
- Pydantic 的 exclude_unset=True 只保留 model_fields_set 中显式设置的字段；默认值字段不在此集合，构造后对默认 dict 的原地修改也不会补记，因此 extra_fields 可能整体缺席，直接索引即抛 KeyError。

2. 核心修复（verl/experimental/agent_loop/agent_loop.py，+5/-2）

- 在序列化输出上执行 output.setdefault("extra_fields", self.extra_fields.copy())：仅当 extra_fields 缺失（默认值场景）时恢复一份浅拷贝；显式传入的实例已存在于序列化结果中，setdefault 不会覆盖调用方数据。
- teacher 字段提升改为在局部变量 extra_fields 上 pop("teacher_ids"/"teacher_logprobs")，缺失时返回 None 而非 KeyError；由于 pop 发生在拷贝或 Pydantic 序列化出的独立映

射上，self.extra_fields 不再被串行化过程改写，序列化保持只读语义。

3. 回归测试（tests/experimental/agent_loop/test_agent_loop_extra_fields_schema_on_cpu.py，+43）

- 新增 _make_agent_loop_output() 工厂与 3 个用例：test_agent_loop_output_as_dict_handles_default_extra_fields（空默认值补出空 dict）、test_agent_loop_output_as_dict_preserves_mutated_default_extra_fields（动态填充字段保留）、test_agent_loop_output_as_dict_promotes_teacher_fields_without_mutating_model（teacher 字段提升且模型自身状态不变）。
- 文件名满足 *_on_cpu.py 模式，会被现有 cpu_unit_tests.yml 自动收集，无需新增 CI 配置或 workflow。

4. 本地与实机验证

- pre-commit run --all-files 全部 13 个 hook 通过（Ruff、mypy、compileall、sanctity 等）。
- 提交者本地跑通 Model Factory CPU agent-loop 回归套件 8 例；Qwen3-4B LangGraph GRPO val_only 在 2 张 H20 上实机验证，MATH acc/mean@1=0.996，完整日志无 KeyError 无 traceback。

关键文件：

- verl/experimental/agent_loop/agent_loop.py（模块 智能体循环；类别 source；类型 core-logic；符号 AgentLoopOutput.as_dict）：修复入口：AgentLoopOutput.as_dict() 用 setdefault 恢复被 exclude_unset 遗漏的默认 extra_fields 浅拷贝，并在拷贝上完成 teacher 字段顶层提升，避免序列化过程污染模型对象；这是所有 AgentLoop 实现共用的数据契约出口。
- tests/experimental/agent_loop/test_agent_loop_extra_fields_schema_on_cpu.py（模块 智能体循环；类别 test；类型 test-coverage；符号 _make_agent_loop_output, test_agent_loop_output_as_dict_handles_default_extra_fields, test_agent_loop_output_as_dict_preserves_mutated_default_extra_fields, test_agent_loop_output_as_dict_promotes_teacher_fields_without_mutating_model）：新增 3 个 CPU 回归用例，分别覆盖空默认值、动态填充默认值、teacher 字段提升且模型状态不变；文件名满足 *_on_cpu.py 模式，会被现有 cpu_unit_tests.yml 自动收集，无需新增 CI 配置。

关键符号：AgentLoopOutput.as_dict, _make_agent_loop_output, test_agent_loop_output_as_dict_handles_default_extra_fields, test_agent_loop_output_as_dict_preserves_mutated_default_extra_fields, test_agent_loop_output_as_dict_promotes_teacher_fields_without_mutating_model

关键源码片段

verl/experimental/agent_loop/agent_loop.py

修复入口：AgentLoopOutput.as_dict() 用 setdefault 恢复被 exclude_unset 遗漏的默认 extra_fields 浅拷贝，并在拷贝上完成 teacher 字段顶层提升，避免序列化过程污染模型对象

；这是所有 AgentLoop 实现共用的数据契约出口。

```
    def as_dict(self) -> dict[str, Any]:
        # exclude_unset=True 只保留 model_fields_set 中显式赋值的字段；
        # 默认值字段不在此集合，构造后对默认 dict 的原地修改也不会补记，
        # 因此依赖默认值的合法实例可能在 output 中整体缺失 extra_fields。
        output = self.model_dump(exclude_unset=True)

        output["prompts"] = torch.tensor(output.pop("prompt_ids"), dtype=torch.int64)
        output["responses"] = torch.tensor(output.pop("response_ids"), dtype=torch.int64)
        output["response_mask"] = torch.tensor(output.pop("response_mask"), dtype=torch.int64)

        response_logprobs = output.pop("response_logprobs", None)
        if response_logprobs is not None:
            output["rollout_log_probs"] = torch.tensor(response_logprobs, dtype=torch.float32)

        routed_experts = output.pop("routed_experts", None)
        if routed_experts is not None:
            # Router replay 按绝对 token 位置索引该字段，必须对齐到完整序列长度；
            # 多轮循环在最后一次生成后停止记录，尾部行保持 0，由 replay 侧 mask 滤除。
            routed_experts = torch.tensor(routed_experts, dtype=torch.int16)
            total_length = output["prompts"].size(0) + output["responses"].size(0)
            aligned = routed_experts.new_zeros((total_length, *routed_experts.shape[1:]))
            num_rows = min(routed_experts.size(0), total_length)
            aligned[:num_rows] = routed_experts[:num_rows]
            output["routed_experts"] = aligned

        # rm_scores: reward score for each token
        reward_score = output.pop("reward_score", None)
        if reward_score is not None:
            rm_scores = torch.zeros_like(output["response_mask"], dtype=torch.float32)
            rm_scores[-1] = reward_score
            output["rm_scores"] = rm_scores

        # 核心修复：仅当序列化结果遗漏 extra_fields（默认值场景）时，用浅拷贝恢复一份
        # 独立映射；显式传入的 extra_fields 已存在于 output 中，setdefault 不会覆盖。
        # 在拷贝上做 teacher 字段 pop，保证序列化过程不污染 self.extra_fields。
        extra_fields = output.setdefault("extra_fields", self.extra_fields.copy())
        teacher_ids, teacher_logprobs = (
            extra_fields.pop("teacher_ids", None),
            extra_fields.pop("teacher_logprobs", None),
        )
        if teacher_ids is not None:
            output["teacher_ids"] = teacher_ids
        if teacher_logprobs is not None:
            output["teacher_logprobs"] = teacher_logprobs
        return output
```

tests/experimental/agent_loop/test_agent_loop_extra_fields_schema_on_cpu.py

新增 3 个 CPU 回归用例，分别覆盖空默认值、动态填充默认值、teacher 字段提升且模型状态不变；文件名满足 *_on_cpu.py 模式，会被现有 cpu_unit_tests.yml 自动收集，无需新增 CI 配置。

```
def _make_agent_loop_output() -> AgentLoopOutput:
    # 所有用例都不显式传 extra_fields，用于复现「依赖默认值」的合法构造路径
    return AgentLoopOutput(
        prompt_ids=[101, 102],
        response_ids=[11, 12],
        response_mask=[1, 1],
        metrics=AgentLoopMetrics(),
    )


def test_agent_loop_output_as_dict_handles_default_extra_fields():
    # 场景一：完全未触碰默认值，序列化结果应补出空 dict
    fields = _make_agent_loop_output().as_dict()

    assert fields["extra_fields"] == {}


def test_agent_loop_output_as_dict_preserves_mutated_default_extra_fields():
    # 场景二：构造后通过可变默认 dict 动态加入字段（修复前会被 exclude_unset 丢失）
    output = _make_agent_loop_output()
    output.extra_fields["raw_prompt"] = [{"role": "user", "content": "hello"}]

    fields = output.as_dict()

    assert fields["extra_fields"] == {"raw_prompt": [{"role": "user", "content": "hello"}]}


def test_agent_loop_output_as_dict_promotes_teacher_fields_without_mutating_model():
    # 场景三：teacher 元数据提升为顶层字段，且提升过程不得污染模型自身的 extra_fields
    output = _make_agent_loop_output()
    output.extra_fields.update(
        {
            "raw_prompt": [{"role": "user", "content": "hello"}],
            "teacher_ids": [201, 202],
            "teacher_logprobs": [-0.1, -0.2],
        }
    )

    fields = output.as_dict()

    assert fields["teacher_ids"] == [201, 202]
    assert fields["teacher_logprobs"] == [-0.1, -0.2]
```

```
assert fields["extra_fields"] == {"raw_prompt": [{"role": "user", "content": "hello"}]}
# 序列化是只读操作：模型内的 teacher 字段必须原样保留
assert output.extra_fields["teacher_ids"] == [201, 202]
assert output.extra_fields["teacher_logprobs"] == [-0.1, -0.2]
```

评论区精华

本 PR 全程无 review 评论，维护者 wuxibin89 直接批准，合并前无追加讨论。实质性的技术论证沉淀在关联 Issue #7486 中，提交者（并用 OpenAI Codex 协助）在 issue 里先完成了根因定位与修复方案设计，PR 按该方案原样落地：

- 方案取舍：setdefault 只对「缺失的默认字段」生效，显式传入的 extra_fields 继续走 Pydantic 的序列化映射，不会覆盖调用方数据；
- 不可变性：恢复的是 self.extra_fields.copy() 浅拷贝，teacher 字段 pop 不会污染原模型对象，串行化仍保持只读语义；
- 范围排除：PR body 明确论证与 #7151（稀疏 reward 元数据）、#7019（output 参数名冲突）不重叠，二者均不触及默认字段序列化，并附上了 issue/ 关键词搜索记录，说明本 PR 并非重复提交。
- extra_fields 默认值在 exclude_unset 序列化中丢失的根因与修法 (correctness)：采纳 issue 提出的窄修复：output.setdefault("extra_fields", self.extra_fields.copy()) 仅在缺失时恢复浅拷贝，teacher 字段提升改为在局部引用上 pop，序列化过程不污染模型状态；显式传入的 extra_fields 继续使用 Pydantic 序列化映射。维护者 wuxibin89 直接批准，无追加讨论。

风险与影响

- 风险：
  1. 浅拷贝边界：self.extra_fields.copy() 只复制顶层 dict。若下游对 as_dict() 返回结果中的嵌套 list/dict 元素做就地修改，仍会回写模型对象；当前仅 pop 顶层键，路径安全，但未来扩展嵌套结构时需留意。
  2. setdefault 依赖「缺失」语义：若某调用路径显式传入 extra_fields=None（schema 不允许但运行时可能出现），setdefault 不会触发恢复，随后 .pop 会在 None 上抛 AttributeError，属于未覆盖的边界。
  3. 测试覆盖范围：新增用例为纯 CPU 单测，未覆盖 TransferQueue 分布式发布路径；提交者以 LangGraph GRPO val_only 实机运行补了端到端证据，但该验证未纳入本项目 CI，后续回归主要靠 3 个单测兜底。
  4. 回归概率低：改动局限在 as_dict() 一个方法、5 行，三个用例覆盖空默认、动态填充、teacher 提升三种关键路径，对显式传 extra_fields 的既有路径零影响。
- 影响：影响面集中在依赖默认 extra_fields 的 AgentLoop 实现：
  - 修复前：TransferQueue AgentLoop（verl/trainer/ppo/v1/agent_loop_tq.py）在 langgraph_agent 等 recipe 的 fallback 轨迹上抛 KeyError，轨迹不发布、prompt 被标记 failed，分布式 GRPO 验证链路直接受损；

- 修复后：默认与动态填充的 extra_fields 均能正确序列化，teacher_ids/teacher_logprobs 提升行为保持不变；
- 显式传 extra_fields 的既有路径零影响，无 API 与配置变更，用户无感升级；团队侧新增的 3 个回归用例把该数据契约行为固化，可防止后续重构再次踩坑。
- 风险标记：Pydantic 默认字段序列化陷阱，浅拷贝共享嵌套对象，仅 CPU 单测覆盖，setdefault 依赖缺失语义

关联脉络

- PR #7422 [rollout] fix: preserve dummy load_format in disaggregated rollout: 同属 rollout 修复线，且此前直接改动 tests/experimental/agent_loop/ 与 tests/experimental/reward_loop/ 下的独立 rollout 测试，与本 PR 的 agent_loop 实验模块回归测试处于同一目录与数据契约范围。
- PR #7151 sparse reward metadata across samples: PR body 明确提及：该 PR 处理稀疏 reward 元数据跨样本问题，与本 PR 的默认字段序列化修复不重叠，同属 AgentLoopOutput 数据契约的邻接改动。
- PR #7019 output argument-name collision: PR body 明确提及：该 PR 处理 output 参数名冲突，不触及默认字段序列化，与本 PR 无交集，用于排除重复提交。