

PR #3168 完整报告

radixark/miles

ci(lark): report every failing test in a job

合并时间: 2026-09-10 06:17

原文链接: <http://prhub.com.cn/radixark/miles/pull/3168>

执行摘要

- 一句话: 增强 CI Lark 通知, 解析每个失败作业的所有失败测试并逐项分析。
- 推荐动作: 此 PR 值得精读, 特别是对于参与 CI 基础设施维护或对 AI 辅助故障诊断感兴趣的工程师。核心设计决策在于“利用 CI 套件自身输出的权威性 (failing_tests 列表) 来约束和指导模型分析”, 这比让模型自行推断要可靠得多。实现上, 对验证逻辑的修改 (确保分析项与测试列表严格匹配) 是保证系统健壮性的关键。该变更是在 PR#3152、#3149 等近期 CI 可观测性改进基础上的重要演进, 值得学习其“从日志中提取结构化事实来驱动下游分析”的模式。

功能与动机

根据 PR body 中的描述, 当前 CI 通知卡片的一行对应一个失败作业, 但一个作业可能运行多个测试。当多个测试失败时 (例如 `stage-c-4-gpu-h200 (1)` 作业中的两个测试), 模型只返回一个分析结果, 导致另一个测试 (例如 `test_dsv4_bshd_thd_rollout_parity.py`) 的信息丢失。这正是 2026-09-09 nightly 发生的问题, 该测试是新引入的, 在自身 PR 中未运行, 且存在严重错误。因此, 需要解析作业的失败测试列表, 并为每个测试提供独立分析。

实现拆解

1. 入口变更与数据提取: 在 `.github/workflows/scripts/ci_failure_analysis.py` 中, 新增 `extract_failed_tests` 函数。该函数使用正则表达式 `FAILED_BLOCK_RE` 和 `FAILED_ENTRY_RE` 从作业日志中解析出 `FAILED:` 块内的所有失败测试路径, 并去重、限流 (`HARD_MAX_FAILURES_PER_JOB`)。
2. 核心逻辑重构: 修改 `validate_response` 函数。它现在从作业数据中读取 `failing_tests` 列表。如果列表存在, 则要求模型为每个测试返回一个 `analysis_item`, 且 `test_name` 必须与列表中的项精确匹配 (通过新的 `_validate_named_test` 函数校验)。验证逻辑确保响应中的分析项与 `failing_tests` 列表完全一致 (不多不少)。如果列表为空 (例如 CPU 阶段), 则沿用原有的单分析路径。
3. 数据结构与约束更新: `AnalysisOutcome.reasons` 字段的类型从 `dict[int, JobAnalysis]` 变为 `dict[int, list[JobAnalysis]]`, 以支持一个作业对应多个分析结果。同时, 在验证层新增了两个拒绝消息: “analyses do not cover the tests the suite named” 和 “test name is not one the suite named”, 并将响应 schema 版本升级至 v3, 将最大分析项数从 15 提高到 30。

4. 渲染与测试配套更新：在 `.github/workflows/scripts/lark_notify.py` 中，`list_jobs_md` 函数的循环逻辑从获取单个分析变为遍历分析列表。相应的，`tests/ci/test/test_lark_notify.py` 和 `tests/ci/test/test_ci_failure_analysis.py` 中的测试用例被更新，以适配新的数据结构，并新增了针对多失败测试场景的完整测试覆盖（例如 `test_the_suite_summary_names_every_failing_test_in_a_job`，`test_a_job_with_two_failures_keeps_both_analyses`，`test_a_response_must_cover_exactly_the_tests_the_suite_named`）。
5. 文档与 Prompt 同步：更新了 `.github/workflows/prompts/ci-failure-analysis.md`，指导模型如何处理新的 `failing_tests` 字段，并明确了在作业提供该列表时，应返回一个分析对应一个测试。

关键文件：

- `.github/workflows/scripts/ci_failure_analysis.py`（模块 CI 分析引擎；类别 `infra`；类型 `core-logic`；符号 `extract_failed_tests`，`_validate_named_test`，`validate_response`）：核心分析引擎，实现了从日志解析失败测试列表、重构验证逻辑以支持多测试分析的关键变更。
- `tests/ci/test/test_ci_failure_analysis.py`（模块 分析引擎测试；类别 `test`；类型 `test-coverage`；符号 `two_failure_jobs`，`analysis_item`，`validate_two`，`test_the_suite_summary_names_every_failing_test_in_a_job`）：核心测试文件，新增了针对多失败测试场景的完整测试覆盖，验证了解析、验证和响应约束逻辑。
- `.github/workflows/scripts/lark_notify.py`（模块 Lark 通知渲染；类别 `infra`；类型 `infrastructure`；符号 `list_jobs_md`）：负责生成 Lark 通知卡片，修改了渲染循环以适应每个作业可能对应多个分析结果的新数据结构。
- `.github/workflows/policies/ci-failure-response-schema.json`（模块 响应 Schema；类别 `config`；类型 `configuration`）：定义了 AI 模型响应的 `schema`，升级了版本并将最大分析项数从 15 提升至 30，以容纳一个作业可能产生多个分析的新场景。
- `.github/workflows/prompts/ci-failure-analysis.md`（模块 分析 Prompt；类别 `docs`；类型 `documentation`）：指导 AI 模型进行分析的 `prompt`，明确指示模型在收到 `failing_tests` 列表时应如何操作。
- `tests/ci/test/test_lark_notify.py`（模块 Lark 通知测试；类别 `test`；类型 `test-coverage`）：Lark 通知渲染的测试文件，更新了所有构造 `AnalysisOutcome` 的测试用例以使用列表类型。

关键符号：`extract_failed_tests`，`_validate_named_test`，`validate_response`

关键源码片段

`.github/workflows/scripts/ci_failure_analysis.py`

核心分析引擎，实现了从日志解析失败测试列表、重构验证逻辑以支持多测试分析的关键变更。

```
def extract_failed_tests(text: str, limit: int) -> list[str]:
```

```
    """从作业日志中提取套件自身报告的失败测试列表。
```

```
    该函数查找以 'FAILED:' 开头的块，解析其中列出的测试文件路径。
```

```
    返回一个去重且限流的列表，确保后续分析可被精确约束。
```

```
    """
```

```
    block = FAILED_BLOCK_RE.search(text)
```

```

if block is None:
    return []
names: list[str] = []
for match in FAILED_ENTRY_RE.finditer(block.group("body")):
    path = match.group("path")
    # 确保测试名称不重复
    if path not in names:
        names.append(path)
# 应用每作业最大失败数限制
return names[:limit]

```

```

def _validate_named_test(raw: Any, named: list[str]) -> str:
    """当套件已命名失败测试时，验证分析项的 test_name 必须是其中之一。

```

此函数是保证分析结果准确性的关键关卡，防止模型报告不存在或虚构的测试。

"""

```

if not isinstance(raw, str) or raw not in named:
    raise ValueError("test name is not one the suite named")
return raw

```

tests/ci/test/test_ci_failure_analysis.py

核心测试文件，新增了针对多失败测试场景的完整测试覆盖，验证了解析、验证和响应约束逻辑。

模拟包含两个失败测试的作业信息

```

def two_failure_jobs():
    return [
        {
            "job_id": 10,
            "evidence_refs": ["job:10:log:1-2"],
            "failing_tests": ["tests/e2e/a/test_one.py", "tests/e2e/b/test_two.py"],
        }
    ]

```

为指定测试名称构造一个分析项

```

def analysis_item(test_name, **overrides):
    item = {
        "job_id": 10,
        "tags": [],
        "test_name": test_name,
        "reason": "Collection failed because the module is gone.",
        "category": "test_failure",
        "confidence": "high",
        "evidence_refs": ["job:10:log:1-2"],
        "related_pull_request": None,
    }
    item.update(overrides)
    return item

```

```
# 验证双失败作业场景
# 此测试确保模型为两个失败测试分别返回分析后，两个分析都能被保留。
def test_a_job_with_two_failures_keeps_both_analyses():
    analyses = validate_two([
        analysis_item("tests/e2e/a/test_one.py"),
        analysis_item("tests/e2e/b/test_two.py")
    ])
    # 断言作业 10 的分析列表包含两个测试名称，且顺序与输入一致
    assert [a.test_name for a in analyses[10]] == [
        "tests/e2e/a/test_one.py",
        "tests/e2e/b/test_two.py"
    ]
```

评论区精华

无实质性讨论。PR 获得了一个来自 claude[bot] 的自动化代码审查提示，但未引发任何技术讨论或设计权衡。

- 暂无高价值评论线程

风险与影响

- 风险：风险较低，主要集中在回归和正确性验证上。1. 回归风险：核心分析逻辑和数据结构（reasons 从 dict 值变为 dict 值的列表）发生了变化，可能影响依赖此数据结构的下游组件（如 Lark 通知渲染）。但所有测试均已更新并通过（845 passed）。2. 正确性风险：新引入的 extract_failed_tests 正则解析依赖 CI 日志的特定格式（FAILED: 块），格式变化可能导致解析失败。PR 作者已用三个真实 nightly 日志验证了解析的可靠性。3. 性能风险：新增的逐作业、逐测试分析可能略微增加模型调用成本和 Lark 卡片长度，但已通过 HARD_MAX_FAILURES_PER_JOB 和 schema 的 maxItems 进行了限制。
- 影响：影响范围明确且集中于 CI/CD 和团队协作流程。对用户（开发者）而言，CI 失败通知卡片现在能提供更精确、更完整的失败信息，直接提升了定位和修复问题的效率。对系统而言，增强了 CI 管道的可观测性和自动化诊断能力。对团队而言，减少了因信息遗漏导致的调试时间，尤其是在处理跨 PR 引入的测试失败时。变更本身不影响核心训练或推理功能。
- 风险标记：CI 日志解析依赖特定格式，核心数据结构变更影响下游渲染

关联脉络

- PR #3152 ci(lark): explain each failed job on the CI status card: 本 PR 是 PR#3152 引入的 AI 驱动 CI 失败分析功能的重大增强。PR#3152 实现了每个作业一行的基本分析，而本 PR 将其扩展为每个失败测试一行。
- PR #3149 ci: repeat a failed test's last output in the suite summary: 本 PR 解析的失败测试列表正是由测试套件打印的摘要块提供的。PR#3149 增强了该摘要块，使其在失败时重复显示测试的最后输出，为本 PR 的分析提供了更丰富的上下文。
- PR #2038 [feat] Enable THD packed-sequence training for DeepSeek-V4-Flash, with context parallelism: 本 PR 的动机案例直接关联：PR#2038 引入的 test_dsv4_bshd_thd_rollout_parity.py 测试在自身 PR 中未运行，导致其在 nightly 中失

败时，旧版 CI 通知遗漏了它。本 PR 的变更正是为了确保此类测试失败不会再次被忽略。