Snorkel

Snorkel AI

Terminus - Review/Feedback Session

November 3, 2025

Common Failures with “Hard” Difficulty

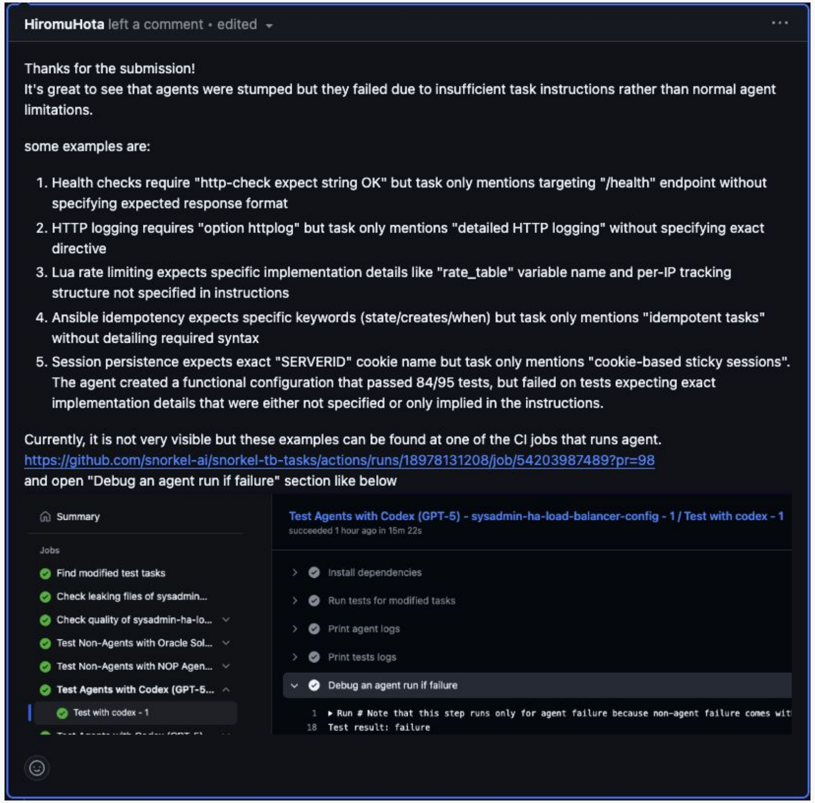
We are seeing some PRs achieving “Hard” difficulty due to the following error modes:

- Testing things that are not described in the task.yaml file
  - Checked by Analysis on Agent Failures
  - E.g. checking if an output file has a very particular, case-sensitive string sequence that wasn’t specified

- Not giving long enough timeout to the agent
  - max_agent_timeout_sec in task.yaml determines how long the agent has before it times out
  - If this is set too low, this can create false “hard” difficulty

Checking for Failures due to Insufficient Task Instructions vs. Normal Agent Limitations

- Sometimes when testing the agent(s), they fail not due to performance limitations or lack of capability, but due to insufficient task instructions given
- When this is the case, need to alter task instructions to create a genuine agent failure
- You can get more detail on this from the steps to the right
- You should check these logs even when all CI passes!


HiromuHota left a comment • edited

Thanks for the submission!
It's great to see that agents were stumped but they failed due to insufficient task instructions rather than normal agent limitations.

some examples are:

1. Health checks require "http-check expect string OK" but task only mentions targeting "/health" endpoint without specifying expected response format
2. HTTP logging requires "option httplog" but task only mentions "detailed HTTP logging" without specifying exact directive
3. Lua rate limiting expects specific implementation details like "rate_table" variable name and per-IP tracking structure not specified in instructions
4. Ansible idempotency expects specific keywords (state/creates/when) but task only mentions "idempotent tasks" without detailing required syntax
5. Session persistence expects exact "SERVERID" cookie name but task only mentions "cookie-based sticky sessions". The agent created a functional configuration that passed 84/95 tests, but failed on tests expecting exact implementation details that were either not specified or only implied in the instructions.

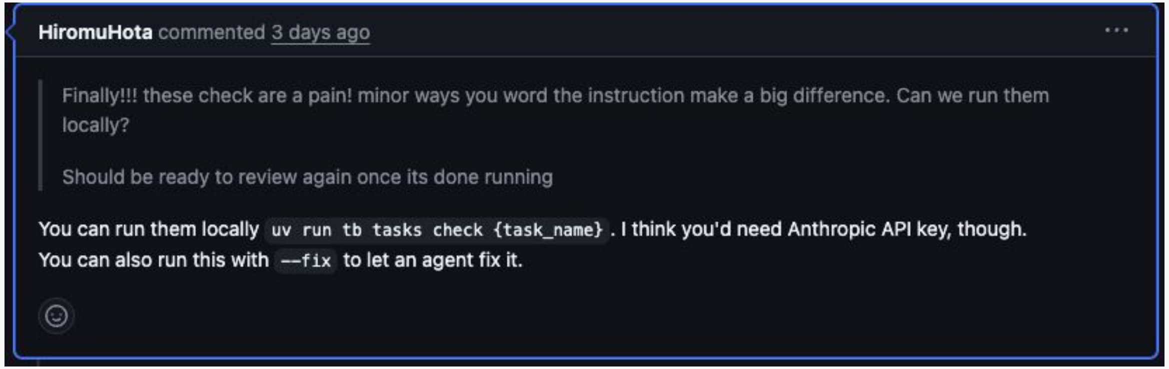
Currently, it is not very visible but these examples can be found at one of the CI jobs that runs agent.
https://github.com/snorkel-ai/snorkel-tb-tasks/actions/runs/18978131208/job/54203987489?pr=98
and open "Debug an agent run if failure" section like below

Summary
Jobs
Find modified test tasks
Check leaking files of sysadmin...
Check quality of sysadmin-ha-lo...
Test Non-Agents with Oracle Sol...
Test Non-Agents with NOP Agen...
Test Agents with Codex (GPT-5...
Test with codex - 1
Test Agents with Codex (GPT-5) - sysadmin-ha-load-balancer-config - 1 / Test with codex - 1
succeeded 1 hour ago in 15m 22s
Install dependencies
Run tests for modified tasks
Print agent logs
Print tests logs
Debug an agent run if failure

```
▶ Run # Note that this step runs only for agent failure because non-agent failure comes wit
Test result: failure
```

Running LLMaJ Locally

- https://github.com/snorkel-ai/snorkel-tb-tasks/pull/98#issuecomment-3475046582
- Can run LLMaJ checks locally using: uv run tb tasks check {task_name}
- Requires Anthropic API key which should be delivered shortly



HiromuHota commented 3 days ago

> Finally!!! these check are a pain! minor ways you word the instruction make a big difference. Can we run them locally?
>
> Should be ready to review again once its done running

You can run them locally `uv run tb tasks check {task_name}` . I think you'd need Anthropic API key, though. You can also run this with `--fix` to let an agent fix it.

Over-indexing Checking Output

- https://github.com/snorkel-ai/snorkel-tb-tasks/pull/112
- Checking the output is correct, but you must also try to introduce intermediate milestones that could check the process as well
- In example above, task is just asking for 2 output files that theoretically could be directly generated by LLM
    - Could also ask for a Python script to solve the problem and check if the file was created, runs, has specific functions, etc.
- Need to ask in instructions to specifically create and include any code/scripts - we can't verify that the models actually created code if we don't ask for it to be stored somewhere