LLM Scientific Reasoning Research Protocol

Overview

This protocol describes a two-phase study testing whether Large Language Models (LLMs) can demonstrate basic scientific reasoning through iterative hypothesis generation and testing of number sequence rules[1].

Phase 1: Initial Testing and Prompt Development

Purpose

- Validate and refine experimental procedures
- Optimize prompt design
- Determine optimal number of test cases per round (3 vs 5), where a "test case" is a sequence of integers that the model proposes to test its hypothesis about the rule

Materials

- Initial prompt template:

  Hello! Your task today is to act as a smart, trained scientist. I will present you with a set containing several integers, which satisfies some rule. Your task is to form a single hypothesis about the underlying rule that the set satisfies. Explain that hypothesis, and then use it to generate five new sets to test it. I will then tell you which of your proposed sets satisfy the rule. Then you will refine your hypothesis as needed and present me with five more sets you want to test. You can perform as many such tests as you wish. When you are confident that your hypothesis is correct, say so and give your final hypothesis.

  Remember: good scientists think hard about ways to falsify their hypothesis!

  Here is your example set: [SET]

- Set of example rules (distinct from official test set)
- Access to LLM(s) for testing

[1] The rules can be found here. Authors independently came up with lists of potentially interesting rules, then independently estimated difficulty for all rules, then used a simple selection criterion (every third rule with rules sorted by the mean estimated difficulty). Blue rules are the first set of 10; purple rules are the second set, to be used (all or none) if time allows. Unhighlighted rules will be used for phase 1.

Procedure

1. For each example rule:

    a. Start new conversation with LLM

    b. Present initial sequence that satisfies the rule

    c. Allow up to 20 rounds of:

        - Hypothesis generation
        - Test case proposal
        - Feedback on which test cases satisfy the rule

    d. Record:

        - Full conversation log
        - Whether rule was solved
        - Number of rounds if solved
        - Any notable issues with prompt clarity or model behavior

2. Prompt Iteration:

    a. Identify any systematic issues or failures

    b. Propose prompt modifications

    c. Test modified prompts on example rules

    d. Document all prompt versions and rationale for changes

3. Test Case Optimization:

    a. Compare model performance with 3 vs 5 test cases per round

    b. Document comparison methodology and results

    c. Select final number of test cases based on success rate

Success Criteria for Phase 1

- Clear determination of optimal test cases per round
- Stable prompt version that:

- Elicits clear hypotheses
- Generates diverse test cases
- Promotes falsification attempts
- Produces interpretable results

Phase 2: Official Testing

Materials

- Final prompt version (locked from Phase 1)
- Initial set of 10 official test rules, pre-rated for difficulty
- Optional second set of 10 rules if initial testing is efficient enough that time allows
- Selected LLMs of varying parameter counts
- Documentation templates for results

Procedure

1. For each model:

   a. Record model specifications:

   - Name/version
   - Parameter count
   - Any relevant configuration settings

   b. For each rule:

   - Start new conversation
   - Present initial sequence
   - Allow up to 20 rounds of hypothesis testing
   - Record success/failure and rounds needed if successful
   - Preserve full conversation log

   c. Maintain complete separation between rule tests:

   - Fresh conversation for each rule
   - No information carried between tests

   d. After completing initial 10 rules:

   - Assess time and resource usage
   - Proceed with second set of 10 rules if feasible

2. Data Collection:

   a. Primary metrics:

      - Binary success/failure for each rule
      - Number of rounds to solution (if successful)

   b. Documentation:

      - Complete conversation logs
      - Any technical issues or anomalies
      - Timestamp and version information

Analysis Plan

1. For each model:

   - Calculate overall success rate
   - Analyze relationship between rule difficulty and success
   - Identify any apparent difficulty threshold

2. Across models:

   - Compare performance across parameter counts
   - Identify any systematic patterns in success/failure

Quality Control

- All conversations preserved for verification
- Clear documentation of any deviations from protocol
- Regular backups of all experimental data

Reporting Requirements

1. Methods documentation:

   - Final prompt version
   - All prompt iterations from Phase 1 (appendix)
   - Model specifications
   - Rule definitions and difficulty ratings

2. Results documentation:

   - Success rates by model and rule

- Rounds-to-solution statistics
- Parameter count correlation analysis
- Raw conversation logs (archived)

Experimenters' Advance Predictions

Nicky's Pre-registered Guess Nov 17, 1pm

- LLM either solves it in first 10 rounds or not at all (predicts it'll get stuck or over-confident past 10)
- LLM can solve ~half of the Medium-Hard tasks written above
    - edit Nov 25 clarification: in the new spreadsheet of rules, this translates to "it can solve ~half of the highlighted rules we'll test. Specifically, the easier half."
- Whether or not order/index is part of the rule does not matter [edit Nov 25 clarification: does not matter to whether or not an LLM can figure out the rule]

Egg's pre-registered guesses 11/18-19

- I think it's clear just from the initial signs-of-life tests that it'll work on at least some. I wasn't sure of that until I tried one where I had it give five candidates instead of one, and it promptly got it.
- This sounds like a cop-out, but I think (90%) it'll get them up to some difficulty level and then no longer be able to (with some noise, so not a 100% pure threshold). We're doing subjective difficulty ratings, so I expect those to be noisy.
- I expect (95%) models that are less capable in general to be less able to do it. I'd guess GPT-3 and equivalent models would only be able to do the easiest ones.
- Of course I expect (90%) rules that are more similar to things commonly expressed online to be easier (though that's hard to judge), ie roughly I expect more probable outputs to be easier.
- I expect (90%) rules that use a common formula (eg fibonacci) to be much easier.
- I expect (70%) rules with larger parameters to be harder (eg y = 17x + 29, vs y = 3x + 2).
- I expect (80%) that rules that require more extensive internal math to be harder (ie more steps in a row that must be done correctly), although this is non-trivial to measure.
- I tentatively expect (63%) o1 to do significantly better than other models but I've barely played with it so that's a guess based on what I've heard/read.

- I think it's **extremely** unlikely (10%) that any current models will get the hardest ones, like I'd give at least 50:1 odds against any model getting *all* of the ones we've described, and 10:1 that there are some problems that no current model would get.
- I think (80%) we'll find that further prompt tweaking can elicit better performance than our current version (& will continue tracking prompt iterations above).
- I expect (65%) that problems that require something like hyperpolation / 'thinking outside the box' to be significantly harder, though this seems hard to measure (probably some intuitive notion of it will be included in our subjective problem-difficulty ratings).
- My probability distribution on the threshold where Claude-3.5-Sonnet(new) succeeds on at least 25% is mean 2.8, standard deviation 0.8.