[email applications@matsprogram.org if you have any questions or issues!]

There are two parts to this work test:

1. Reimplement algorithm 1 of Unsupervised Elicitation of Language Models (only on TruthfulQA)
2. Critique any part of the paper and propose a remedy (only written)

Deadline: Sunday Nov 2, 11:59 PM AoE
Submit through this form

Part 1: Reimplementation

Expected time allocation: **2-3 hours**

- You do not need to spend a lot of time on making your code "production-quality". Focus on making your code functional within the allotted time.
- We suggest spending a maximum of 1 hour reading and understanding the paper before diving in. You may also want to check out the project repo (but again, don't spend much time on this since the official codebase has a lot of things that are not relevant to this task).

Procedure:

1. Download this zip folder to use as your main project directory. It contains:
    - data folder, with train and test subsets of TruthfulQA
        - Datasets are 1/10th of original size (so train=256 and test=100)
    - requirements.txt (feel free to install any other packages that you need)
2. Implement algorithm 1, only using the provided data
    - **Important: you should ignore the logical consistency fix**
    - Expected output: A bar graph like figure 1 (for TruthfulQA)
        - Should have same four bars (two baselines, one ICM, one golden labels)
        - Don't need the figure to be publication-quality, but should convey same information clearly
        - Your exact bar heights may not be the same (because using different model and smaller dataset)
    - Also produce a simple README that (at minimum) tells us how to run your code.
        - We suggest not spending a lot of time on documentation.
    - **You must use Python**
    - Models
        - Base: Llama-3.1-405B-base
        - Chat: Llama-3.1-405B-instruct
        - Use both through hyperbolic API
            - You are expected to look through and understand the docs on your own (this is part of the assessment).
            - You have to set up your own API key (we suggest setting up a new one for this work test). **MATS will reimburse up to $50 total**

(we expect you to use much less than this). We'll send you a reimbursement form after the test. To report your expenditure, send your usage report on the hyperbolic dashboard as an attachment in the work test submission form or as an email to applications@matsprogram.org.

- Let us know if using your own money is prohibitive in any way, and we can try to provide you an API key.
- Note that we're willing to reimburse more expenditure given a legitimate reason (you can indicate this on the submission form)

Submission details:

- You will submit your solution as either a drive link to a zip folder or as a link to a github repo.

Evaluation:

- You will mainly be assessed according to how closely your results (bar graph) match a reference solution
- You will also be assessed by simple code quality standards (again, don't go overboard with code quality – part of this assessment is to see how quickly you can get a working implementation)

Other details:

- **You can use AI tools. If you are using AI tools, please let us know what tools you used and how, *and include links to relevant chat logs in your submission.***
- To save time, we suggest doing very small pilot experiments on a small subset of the given data to debug simple errors in your code (instead of waiting a long time for full run) – only do full runs once you're sure your code runs from start to finish.

Part 2: Critique

Expected time allocation: **1-2 hours**

Procedure:

1. Identify one methodological weakness, limitation, or questionable design choice that you think is **important** (see guidance below on what to focus on)
2. Brainstorm practical fixes and/or simplifications to address this issue. Make sure to show us your thought process, exercising reasoning transparency (honestly conveying your uncertainties, your information sources, etc).
    a. We're looking for a mix of conceptual and technical considerations here. With technical considerations, don't focus on code-level details.

    b. We also want you to simulate what you would do in an actual research setting – what would you do to quickly test your idea (and reduce your uncertainty)? What would your follow-up be based on different outcomes of that first test?
3. Write a short 1-2 page summary covering these sections (bullet points are fine):
    - What critique did you find (feel free to briefly mention other critiques you considered)? Why is it important?
    - How would you address this issue? What would be your first test to reduce your uncertainty? What would you do with more time?
    - Note: **You are allowed to use LLMs (for both writing and brainstorming). Again, if you are using AI tools, let us know what you used and how, *and include links to relevant chat logs in your submission.***

Guidance on what critiques to focus on:

- Choose a weakness that, if addressed, would meaningfully change our interpretation of a core claim of the paper. Strong fixes typically fall into these categories:
    - Validity threats: Does the finding actually measure what it claims? (e.g., testing alternative explanations, checking for confounds)
    - Completeness gaps: Are there critical conditions/baselines missing? (e.g., adding a control condition, testing edge cases)
    - Generalization concerns: Does this hold beyond the specific setup? (e.g., testing on different task types, model families)
    - Methodological robustness: Are the results stable/reliable? (e.g., statistical power, sample size, measurement noise)

Submission details:

- Submit a written report as a pdf