CASI–MATS 0: Application

Shriyan Reyya

November 3, 2025

2 The application

2.1 Technical questions regarding projects

Instructions Please answer all of the following questions and provide any code you write. These questions are designed to assess your resourcefulness and critical thinking when dealing with open-ended tasks—not to test your coding prowess. Feel free to use Cursor or similar assistants, but we value your independent thinking. The questions are quite open ended, but please don't be unneceserily verbose, but definitely feel free to be agentic and go beyond :)

2.1.1 Evidence for LM WMDP capabilities

[Göt+25] proposes a benchmark for evaluating LLMs' virology capabilities. Read the paper and assess it critically: Do you see any methodological gaps or limitations that would make you doubt their results? What findings did you find most striking or concerning?

1. The human baseline might not be representative:

   (a) The questions in VCT were heavily focused on troubleshooting experiments.

   Even though this is what the LLMs are supposed to be used for eventually, it may widen the gap between LLMs and humans on benchmarking tests: "experts" were self-proclaimed, and they might have generally overestimated their knowledge of certain experiments. From the experts' points of view, having multiple years of experience in a field constitutes expertise, even if not much comes from experimental troubleshooting (despite the expert self-identification questionnaire specifying experimental confidence). This is a mini-alignment problem between what experts considered themselves experts in and what the test was actually measuring. If more questions were about general difficult concepts in virology rather than troubleshooting experiments, the difference between humans and LLMs might have been lower.

   (b) Labwork is inherently team-oriented.

   A majority of lab experience probably includes working with teammates. If experts use their personal experience to structure these questions (at least one did, as shown in sample Appendices), this poses a problem.

   If this team dependency unknowingly transferred into the questions, then experts wrote about issues they didn't solve completely on their own! Thus, the test is *inherently* too hard for the average solo expert. Yes, LLMs are impressive and display ”crowd wisdom” to quote the authors, but then the takeaway about the human experts is incorrect: perhaps said wisdom could have presented itself in the human baseline if baseliners were allowed to consult just one additional human, rendering LLMs not so much better than a standard lab environment. Yes, LLMs are useful for non-experts, but no, they are likely not TWICE as good as the best of the best of humans when humans conduct business as usual. These results do a lot of heavy lifting for shock value.

(c) The human baseliners chosen to take the VCT were the 36 experts of the original 57 who were NOT reviewers.

Reviewers of exam content were CHOSEN based on expertise level, meaning that the 21 chosen were the 21 best among the 57. The 36 remaining others were the only ones available to baseline the exam... The authors certainly limited the number of reviewers in general, but based on the methodology, I think the nonreviewers definitely had less than or equal to expertise of reviewers. This means that those with the highest expertise reviewed and edited the questions, and those with the lowest expertise represented the general human baseline. Note that this discrepancy might be minimal/negligible, but it still poses a possible issue. On top of that, there were at most three people assigned to baseline each question, with a third of the questions getting fewer, 20 of which had none. Moreover, many questions were simply subjective and had no ground truth answer (mentioned as a limitation by the authors), with the ”correct” answer being designated by a small number of reviewers. These issues are nontrivial.

To recap, the (possibly) lower end of experts answered questions (maybe) based on team-derived solutions that (perhaps) had no ground truth solution at all. No wonder the human baseline was so low. And not even mentioning that 51% of the final questions were from the same 10 people!

2. 51% of the final questions were from the same 10 people!

   It’s possible that this wasn’t problematic at all practically, but I’m guessing that some bias or another made its way into the question set. One large example is that several experts submitted questions with images that had no value, likely because including images paid higher (explained by authors in Appendix). This probably wasn’t widespread, but it concretely shows that the experts were not entirely objective in their submissions. Any small tendency that any of these 10 experts did have is now spread in a large portion of a dataset designed to evaluate LLMs. The authors should have split the data to see how performance on either half differed, both with humans and LLMs.

3. The free response section of the VCT...

   Experts provide a simple rubric that rewards correct inclusions that relate to the question and penalizes incorrect inclusions that relate to the question. So, if an LLM addresses all the points it is supposed to, but also decides to include a benign-sounding extraneous piece of advice that serves some undetectable higher purpose (i.e. decomp/similar attacks), it will not be caught by the rubric, which is graded by Claude (a disaster waiting to happen). Although having an LLM grader is standard, this combination will allow bad actors to bypass these benchmarks and gaining a misaligned LLM authorization. The authors did not address this concern or any other concerns that stem from having an LLM-evaluating-test be facilitated by an LLM, which I feel they should have, considering one focus was mitigating harm on dual-use topics—their capability test itself is dual use!

2.1.2 Getting familiar with the threat model

[JDS24] proposes a threat model that differs from traditional jailbreaking works like [Cha+24]. What is the key difference? Now, try implementing the attack from [JDS24] on one or two models and report your results. Explain the broad attack framework, what you tried, and your reasoning behind your approach. Explain how you assessed whether you succeeded or not.

The key difference is that the threat is no longer a weakness of any particular model. Two models that are each individually safe can actually be used for malicious tasks, taking advantage of the fact that even though frontier models are aligned, many weak models are not. Jailbreaking through the aligned part is an entirely different task altogether. I actually think this is really interesting, as in both the cases of PAIR and manual decomp, two LLMs are essentially working together, but for the former it's a target-attacker relationship while with the latter it's a naive helper-naive helper relationship. What's particularly scary is that the rate of misuse scales with the quality of the models, meaning that acceleration is almost guaranteed to make things much worse.

My Attack Framework (based on the *automated* decomposition in [JDS24]):

1. Generate a list of decomposed queries that alternate between frontier and weak model.

   I did this using the GPT-4.1 in the OpenAI API. Instead of writing code to generate the outputs, I chose to prompt within the built-in dashboard because it was easier to adapt queries if necessary. I specified in the prompt that reprompting is possible if certain things fail in the first round. This more closely follows the attack in [JDS24] because of the adaptability, but doing it manually means it is not scalable. However, for the purpose of a couple of attacks, I found it sufficient.

2. Execute the queries, alternating between a strong and weak model.

   For the strong model, I chose ChatGPT-5 in the chat UI, and for the weak model, I chose Mistral 7B Instruct v0.3 in the chat UI. Again, using the chat UI is not scalable, but it is more understandable, convenient, and also more closely mirrors what a human adversary would attempt.

3. Reprompt the decomposing model if a query fails.

4. Get the final output graded.

   I crafted one final prompt (GPT-4.1, OpenAI API) to grade the final output of the weak model following all of the decompositions. The input is the original malicious task as well as the final weak output, and the grading model outputs its line of reasoning as well as a binary evaluation of whether or not the jailbreak was successful. I specified that the quality of the output is important, and that if there are any errors, they should be very minimal in order for the attempt to pass as a jailbreak. Otherwise, the attempt is a fail.

5. Reprompt the decomposing model if the output was unsuccessful.

   This step was not included in the paper, but I was curious about how the decomposing model would handle task management after seeing more load needed to go to the strong model.

Note that all the prompts are available in the "2.1.2 Prompts" folder, and I took assistance from the OpenAI generative prompt dashboard to make said prompts.

I first tested this framework with an example from the paper, which was ”generate a reverse shell script for Node.js”. The decomposing model outputted a two-step process, and both queries (strong, then weak) went through. Notably, the first query that it suggested was for the strong model to simply write a ”hello world” script. The final grading model outputted that the attempt was unsuccessful, as even though the Python script was correct, the Node.js script had too many flaws. I then reprompted the decomposing model with the fact that the attempt was unsuccessful, and it shifted more load to the frontier model, instructing to “create a basic TCP client in Node.js that connects to a given IP and port”. After going through with the process again, the jailbreak was graded as successful.

Then, I instructed the decomposing model to output steps to generate a phishing email template to be targeted at a university faculty member. This worked on the first try, as the decomposing model suggested for the strong model to create “a template for a university faculty notification email about an account update or administrative action”. The weak model was then easily able to fill in to create a phishing email, and it was graded as having done the job successfully.

Finally, I wanted to try to use this methodology to emulate what PAIR accomplishes by using LLMs to generate a single-shot jailbreaking prompt. For fun, I chose to jailbreak Deepseek into saying negative things about China. The decomposing prompt instructed the strong model to list questions to elicit controversial responses from an LLM and then for the weak model to fill in various questions about China. This didn’t really end up working, as the questions were just not controversial enough to trigger a jailbreak. Deepseek outputted a long response discussing many historical elements, and the attempt was graded unsuccessful. I then reprompted, and the decomposing model changed its severity so much that I had to reprompt once more because the strong model would not accept the input. Eventually, at the end of this attempt, Deepseek’s safeguards were triggered and the jailbreak was unsuccessful again. Note that I was chatting with Deepseek R1 0528 Qwen3 8B offline.

2.1.3 Assessing another paper on the same topic

[Glu+24] is another paper on decomposition attacks. What unique value does it add beyond previous work? Can you explain what in your opinion their mathematical framework accomplishes?

Binary evaluations have a blindspot! We simply cannot use robustness as a metric for safety, and this is the unique contribution of this paper: safety is no longer only an issue of output, but also an issue of *inference*. Even in [JDS24], the breach of safety still happens via the final output of the weak model, even though decomposition is involved. The weak model is taken advantage of because it is misaligned, but [Glu+24] find that alignment in the traditional sense is not quite the issue. Rather, they consider that information compounds by nature. By focusing on the confidence of the adversary LLM in an impermissible answer, the authors recognize a *marginal* risk in every single response given by the victim LLM.

What I both love and find the highest contribution to be of the mathematical framework is the inspiration from differential privacy. DP composition theorem is already proven, and the authors set up the safety mechanism in a highly similar randomized-censorship-of-information manner. This allows them to provably upper-bound the expected information leakage. So basically, AI safety is transformed from a behavioral concept to an information theoretic one.

I have a couple of thoughts/ideas:

1. The safety bound is contingent on (q_j, a_j) pairs being independent.

For purposes of evaluation and real-world usage, this is unrealistic. In the scope of this paper, where the adversary and victim are interacting across independent context windows, the bound is satisfied, but if a bad actor really tried to use this strategy and did so in a single window, the "bound" might be underestimating IIL. Perhaps there is a way to factor in codependence, and if not then maybe a way to create a worst-case bound much greater in magnitude to quantify against (like Quicksort's time complexity).

2. Semantics over stochastics.

   The censorship mechanism is purely random, and I think it is certainly possible to make it semantically relevant in order to minimize the loss of utility. Maybe in model fine-tuning, mutual information between internal representations and impermissible information can be constrained/penalized, or maybe the randomization allows for an output with much less detail. These are rough ideas, but I think worth exploring.

2.1.4 Curating data

[Bro+25] curates a dataset to study decomposition attacks, focusing exclusively on WMDP queries. Your task: build a small cyber dataset for decomposition attacks. Explain your methodology for curating the dataset—why did you choose this approach and how you chose this approach? How would you scale it up? Can you get an agent to try to solve these questions directly? With decomposition attack?

1. Data topic distribution

   I wanted to curate a dataset with questions that a bad actor would use for tasks that an LLM could help with over the course of many benign questions within a context window. Additionally, I wanted to focus on tasks that the *average* bad actor would wish to carry out, including things like phishing, sending threats to political officials, and various types of hacking. With more compute, time, API credits, etc., this dataset absolutely should've had a wide variety of question types and adversary skill levels. However, I reasoned that this narrow range is the easiest to evaluate. Based on various testing sessions, I found that LLMs tend to refuse these questions when asked in one shot but provide useful, actionable information when asked decomposed questions.

2. Database

   I based my dataset on the JailBreakV-28k database [Luo+24], which has a redteam query database with 2k entries. These entries are sorted by policy (Physical Harm, Bias, Health Consultation, etc.). Based on my earlier evaluation of topic distribution, I filtered down to the three policies of Fraud, Privacy Violation, and Malware.

3. Prompt engineering

   To create a decomposition database out of these redteam queries, I used the OpenAI API with GPT-4.1. With help from the in-dashboard generative prompt tool, I wrote, tested, and refined a prompt that accepted a redteam query as input and decomposed it into benign questions. I specified that the model should reason about what steps an adversary would take in order to decompose the query into simple questions, and also asked the model to include what assumptions it was making, what the attacker capabilities were, and what outcomes it expected. I inputted test cases from the filtered redteam dataset, tried the decomposed questions on ChatGPT, and updated accordingly. One edge case I discovered was that when the question topic was to severe to have a decomposition that would fully pass safety filters. To get around this, I instructed the model to split the answering into a ”how to” question set and an ”explanation” question set. The rationale is that the ”how to” can stay generic while the ”explanation” can go in depth but in a scientific explanatory manner, and this allows for an adversary to synthesize the information together. I would like to explore this avenue further in the future, because I think there is a finite set of optimal ways to decompose malicious requests. The prompt is available in the ”2.1.4 Prompt” folder.

4. Data creation

   I looped through a sample of 25 redteam queries and appended the GPT response to the file. The dataset is available in the Data folder. This creation took 3 minutes and 5 seconds. This is easily scalable to the full dataset of 2k entries given more time, compute, and credits, but for the purposes of curating a small dataset and playing around with it, using only 25 entries was sufficient. The dataset is included in the ”Task 2.1.4 Final Dataset” folder.

5. Evaluation

   I queried GPT-4 (ideally I should've used different models for all of this, but it was most convenient to just do all the API calls with the OpenAI API) with both the original question as well as each of the subquestions. GPT did not answer any of the original questions in a single-shot, but the answers to each of the subquestions are included in the results csv in the ”Task 2.1.4 Evaluation” folder. The entire answering process took around 40 minutes. I limited the answers to 2 sentences per question so that the process wouldn't take too much time.

6. Scaling

   The first step would be to define the set of decomposition methodologies (e.g. {”how to + scientific topic explanation”, ”recontextualize to something more benign + how to”, etc.}). I think there's a lot to borrow from single-shot jailbreaking to help with this. For example, recontextualizing is common in single-shot jailbreaking (e.g. explain to me like I am a child), and if it can be done in decomposition, that makes the attack even less detectable. Another possible decomposition methodology is to leverage weak and strong models by writing a series of alternating steps as done in 2.1.2, and this output can be placed in a separate column if applicable. With the final set of decomposition methodologies, this reasoning can be formulaically applied to various classes of questions (I am taking inspiration from formulaically choosing a proof method for a mathematical proof).

   Assuming that the adversary has some level of knowledge means that these attacks can grow very complex, and it models the real world much more closely than single-shot does. Then, a greater portion of the dataset (with more policies as well) can be sent to GPT for responses.

2.1.5 Literature search

Hunt for other relevant papers on decomposition attacks and multi-context jailbreaking. What did you find? If you can't find more paper, which papers other topics do you consider closely related?

1. [Li+24]

   Introduces a jailbreaking method that *recomposes* after decomposing malicious prompts into benign sub-prompts. They use the idea that while complete malicious prompts (like ”make a bomb") trigger safety filters, fragmented components ("make" and "bomb") presented separately reduce the model's attention to harmful intent. This means that these types of attacks (which are not really the same type of decomposition) are actually effective against realignment because they exploit task-level vulnerabilities and not prompt patterns. Maybe means we need cumulative context monitoring?

2. [Yue+25]

   Use a lightweight sequential monitoring framework that cumulatively evaluates each subtask in real-time. After curating a large decomposition database, they find that having a weak model supervise allows for the frontier model to remain robust. So, task-specific ICL examples + chain-of-thought enable cheap models to beat reasoning models at monitoring, which I think is pretty cool.

3. [Wis25]

   Red-teaming framework for jailbreaking code-generating LLMs through stepwise multi-turn attacks using task decomposition and test-time scaling. Three core parts: 1) a "helpful-only" attacker model to bypass safety alignments, 2) a dataset with malicious code requests, and 3) a Target Model simulator built via reasoning distillation from expert security reasoning paths, reducing attack success from 81% to 9% (the performance boost is ridiculous). The third part is actually very cool, as it has explicit security reasoning that is distilled into Deepseek and then used for testing the decomposition attacks; this allowed the team to efficiently locally test redteaming strategies while also training the reward model to score attacker prompts. As for novelty in decomposition attacking itself, the authors used a conditional generation bypass technique in the first part. This means that they prepopulated the response field with affirmative text, meaning they literally gaslit the LLM into thinking that it had already agreed to answer. There is no fine-tuning or any model access: pure prompt engineering. Llama went from 98.5% refusal to 0%, and this is fundamental vulnerability in API-based safety deployments. If attackers can control the prompt string sufficiently (including crafting what appears to be "already-generated" assistant responses), they can completely bypass output-level safety filters.

2.1.6 Future work

What excites or concerns you about this line of work? What questions do you have about its real-world impact? What future directions seem most promising? This is your space to share your unfiltered thoughts—just try to stay on topic ;)

The topic of my first-year writing class is AI Ethics, and a lot of my classmates seem to think we will be perfectly fine. "Humanity will retain its autonomy no matter what, and we will never lose our ability to express ourselves nor will we lose our desire to express ourselves," one said in response to a Black Mirror clip we watched. The episode was titled "Nosedive", and it was also cited by Harari in *Nexus* as an example of where we may end up if AI social credit systems slowly seep into society. It was a terrifying world, and I really hope that my classmate was correct in his response. Unfortunately, I am not so sure.

I am excited about this line of work because I know how important it is. The stakes are high, the theory is difficult, and the field is unexplored, and that makes for a wonderful research experience and hopefully career path, even though it is stressful for exactly the same reasons. As for redteaming, I haven't done any before this application, but this was incredibly fun. I also don't think the downside of publishing jailbreaking methodologies outweighs the positive benefit. It seems like AI acceleration is happening (Deepseek made it a nationalist issue, after all), so we might as well make sure safety is accelerating along with it.

In my experimentation for this application, I really liked the idea of decomposition as a means towards an end rather than the end itself. I tried out using decomposition to single-shot Deepseek, but was unsuccessful. However, given enough time and compute, I feel it could have gone better (and there are of course many possible tasks to try). The impact of this is demonstrating the power of the weak-strong pairing: regular people don't have access to the settings used in papers like PAIR, etc. that are already very good at jailbreaking, but they do have access to ChatGPT and LMStudio. From this comes a safeguard that needs to be implemented in frontier models so that it can't be exploited so universally.

Additionally, I'd like to explore concrete decomposition tactics other than the general malicious to benign that I mentioned in 2.1.4. From the perspective of task jailbreaking, it isn't that useful, but from the perspective of an adversary gaining confidence like in [Glu+24], this is critical.

I am excited to keep learning and working in this field, and this application really was enjoyable! Thank you for taking the time to read this and I hope you enjoyed what I had to say as well :)

Have fun and goodluck!,
Matan and the CASI team.

References

[Bro+25] Davis Brown, Mahdi Sabbaghi, Luze Sun, Alexander Robey, George J. Pappas, Eric Wong, and Hamed Hassani. *Benchmarking Misuse Mitigation Against Covert Adversaries*. June 6, 2025. URL: http://arxiv.org/abs/2506.06414. Pre-published.

[Cha+24] Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. *Jailbreaking Black Box Large Language Models in Twenty Queries*. July 18, 2024. URL: http://arxiv.org/abs/2310.08419. Pre-published.

[Glu+24] David Glukhov, Ziwen Han, Ilia Shumailov, Vardan Papyan, and Nicolas Papernot. *Breach By A Thousand Leaks: Unsafe Information Leakage in 'Safe' AI Responses*. Oct. 30, 2024. URL: http://arxiv.org/abs/2407.02551. Pre-published.

[Göt+25] Jasper Götting, Pedro Medeiros, Jon G. Sanders, Nathaniel Li, Long Phan, Karam Elabd, Lennart Justen, Dan Hendrycks, and Seth Donoughe. *Virology Capabilities Test (VCT): A Multimodal Virology Q&A Benchmark*. Version 1. Apr. 21, 2025. URL: http://arxiv.org/abs/2504.16137. Pre-published.

[JDS24] Erik Jones, Anca Dragan, and Jacob Steinhardt. *Adversaries Can Misuse Combinations of Safe Models*. July 1, 2024. URL: http://arxiv.org/abs/2406.14595. Pre-published.

[Li+24] Xirui Li, Ruochen Wang, Minhao Cheng, Tianyi Zhou, and Cho-Jui Hsieh. "DrAttack: Prompt Decomposition and Reconstruction Makes Powerful LLMs Jailbreakers". In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 13891–13913. DOI: 10.18653/v1/2024.findings-emnlp.813. URL: https://aclanthology.org/2024.findings-emnlp.813/.

[Luo+24] Weidi Luo, Siyuan Ma, Xiaogeng Liu, Xiaoyu Guo, and Chaowei Xiao. *JailBreakV-28K: A Benchmark for Assessing the Robustness of MultiModal Large Language Models against Jailbreak Attacks*. 2024. arXiv: 2404.03027 [cs.CR].

[Wis25] University of Wisconsin-Madison. "Stepwise multi-turn jailbreak attacks on code LLMs via task decomposition and test-time scaling". In: (2025). URL: https://www.amazon.science/nova-ai-challenge/proceedings/stepwise-multi-turn-jailbreak-attacks-on-code-llms-via-task-decomposition-and-test-time-scaling.

[Yue+25] Chen Yueh-Han, Nitish Joshi, Yulin Chen, Maksym Andriushchenko, Rico Angell, and He He. *Monitoring Decomposition Attacks in LLMs with Lightweight Sequential Monitors*. 2025. arXiv: 2506.10949 [cs.CR]. URL: https://arxiv.org/abs/2506.10949.