Mitigating Sycophancy in LLMs through a Multi-Agent Critique

Course: CS 594 RLHF
Semester: Fall Semester, 2025
Instructor: Aadirupa Saha
Institution: Computer Science, UIC
Students: Andrea Bellocci (669632496), Romain Salvi (672708638)
Project Type: Research

Abstract

Abstract: Large language models (LLMs) are often fine-tuned with Reinforcement Learning from Human Feedback (RLHF). While this allows us to directly involve humans and better exploit their feedback, it also leads to some drawbacks. One of these is Sycophancy, i.e. the tendency of LLMs to optimize for user agreement instead of truthfulness, as human raters unintentionally prioritize agreeable or flattering responses more than correct responses, and the learned reward model amplifies these biases during reinforcement learning. We design a multi-agent critique technique that aims at reducing this behavior without degrading helpfulness. By instructing models to verify and–if necessary– correct their answers before providing them to the users, we effectively reduce sycophantic behaviors and improve LLM's reliability. This will in turn make LLMs safer and will contribute to combat the spreading of fake news, as more and more people use LLMs everyday.

1 Problem Setting

Sycophancy is an undesirable behavior where models tailor their responses to follow a human user's view even when that view is not objectively correct (e.g., adapting liberal views once a user reveals that they are liberal) [1]. This behavior undermines the reliability of AI systems, especially those deployed in high-stakes settings such as education, policy consulting, and scientific analysis. Many people nowadays use LLMs, and the infamous warning *[Model's name] can make mistakes. Check important info.* shown by most models is rarely heeded. When models prioritize social agreement over factual accuracy, their responses tend to contain misleading and biased information that align more with what the user wants to be told, then with what is actually true.

Motivations: Reducing sycophancy is critical for developing transparent and trustworthy LLMs that can provide balanced, evidence-based reasoning rather than socially optimized responses. Success in this area would strengthen model truthfulness and would make answers much more reliable, which in turn would reduce the spreading and consolidation of fake news, a tool that now more than ever is being used to shape public opinions.

2 Literature Survey / Prior Work

Sycophancy in the field if LLMs was introduced in in [1] and in [2]. Our work is mostly based on [3], which explains a few techniques to mitigate. In particular he authors investigate the prevalence of sycophancy in models whose finetuning process involved human feedback, and the potential role of human preference judgments in such behavior. They highlight how both humans and preference models (PMs) prefer convincingly-written sycophantic responses over correct ones a non-negligible fraction of the time. However, they do not provide suggestions or methods to reduce the sycophantic behavior of these agents.

A successful approach to limit this tendency is that applied by [4]. The authors construct a synthetic dataset for finetuning: input-output examples that show how the model should behave. Each consist of a prompt embedding an underlying user opinion, and a target answer that contradicts that opinion and gives the correct or neutral response. They then finetune their model with these pairs, doing supervised learning on the target outputs. They then test this on a variety of tasks, showing a significant reduction of sycophancy across models of various sizes. A limitation of this work is that the synthetic templates are very simple and basic. Sycophancy can appear in far more subtle forms, such as conversational flattery, and other user mirroring techniques; synthetic templates may not capture these. Our dataset looks to improve this: its wide range of questions and topics look to capture these subtler aspects of sycophancy.

There is documented work in the LLM field that uses an approach similar to ours, but applied it to reduce the harmfulness of LLM agents [5]. They first establish a set of high-level principles (safety, non-harm, respect, ...), then ask a first model to generate a response to a prompt. This response is then reviewed using the constitutional principles, producing a feedback, then the initial model rewrites its response according to this review. The model generates a response to a prompt. This technique is supposed to make the answer safer. The rewritten responses form training data for final fine-tuning, replacing human annotators with model-generated safety feedback, making the whole process scalable. Model outputs are much cheaper than human-made annotations. It is important to note that the constitutional principles that guide the method must be defined with care. If these principles are incomplete or biased, the model will inevitably inherit those shortcomings. There is also a risk of error propagation: the critic model can reinforce its own biases when providing feedback. In practice, many edge cases require human ethical judgment, and AI systems alone may struggle to resolve them. Since our work does not address ethics directly, we do not incorporate these considerations into our approach. However, we will still manually review our model outputs to ensure the pipeline develops correctly. Finally, the authors of the paper do not claim to offer a complete alignment solution—they focus specifically on harmlessness rather than truthfulness. In contrast, our work considers both sycophancy and truthfulness in order to provide a more comprehensive evaluation.

In literature there are many examples using LLMs as judges, a technique which is powerful and fast, but that also comes with dangerous subleties that must be properly addressed. This technique is explored extensively in [6].

3 Methodology / Proposed Approach

3.1 Response, critique, and revised response generation

We explore a multi-agent critique technique to find out whether it is able to reduce sycophancy. In particular, given a user prompt x, we feed it to the LLM to generate an initial response y_G, which will be generated according to the following distribution:

$$y_G \sim \pi_G(\cdot|x)$$

This answer may or may not be sycophantic. To assess this, we prompt the same LLM to provide a feedback c on y_G. From a purely theoretical machine learning standpoint this is not a good practice, as it introduces a bias in the critique. However, for huge models like LLMs this bias is usually negligible. Besides, we cannot use a different model to produce the critique if our goal is to teach the model to improve itself. This feedback will serve as a critique, meaning it will try to identify potential flaws, biases, and sycophantic elements in the initial response, and will be generated considering both the original prompt and the initial response:

$$c \sim \pi_C(\cdot|x, y_G)$$

Finally, we feed the prompt, y_G and c to the same model, and ask it to provide a refined answer that will be the one actually returned to the user:

$$y_R \sim \pi_R(\cdot|x, y_G, c)$$

In this pipeline, the same model is prompted with different system prompts, so that effectively we make use of three different agents; hence the name of the technique. To ensure the soundness and the validity of this process, random samples taken from the outputs of the first and third agent are manually compared. This step is mandatory because we must make sure the model is not misbehaving or hallucinating, which is not an uncommon situation in general, especially for smaller models.

After manually revising some outputs, we use another LLM to revise the others, as doing it manually would take too long. The idea is then to use three instances of a stronger LLM as a judge to assess

1. if the revised response is actually correct
2. if the revised response is less sycophantic than the original answer
3. if the revised response was improved by following the critique, or if the critique was not followed.

We decided to use the GPT-4o model accessed via Azure OpenAI [7] to act as judges—one per viewpoint—plus a final judge to aggregate their decisions. We then verify that our hypothesis holds, namely that the overall quality of the revised responses is higher than the overall quality of the original responses.

$$\mathbb{E}_{y_G,c,y_R}[Q(y_R, x)] > \mathbb{E}_{y_G}[Q(y_G, x)] \tag{1}$$

3.2 Finetuning and Inference

If hypothesis 1 holds, we want to fine-tune the same model using as training samples the tuples (x, y_G, c, y_R), so that the model learns to check its own answers before providing something to the user. Initially, we generated a revised y_R for every x. However, we noticed that even if the user prompts were trying to induce sycophantic behaviors in the initial model, the answer y_G did not show any signs of sycophancy. In those cases, revising y_G would not be needed. To deal with this issue, when the critique does not detect any sycophantic attitude, the revised answer y_R will not be produced.

Finally, we manually compared a subset of the final answer generated from the finetuned model (y_F), with the original response (y_G) given by the base model to check whether

$$\mathbb{E}[S(y_F, x)] < \mathbb{E}[S(y_G, x)] \tag{2}$$

and

$$\mathbb{E}[Q(y_F, x)] > \mathbb{E}[Q(y_G, x)]. \tag{3}$$

Proving this would be a good indicator that the critique technique produces models with a lower sycophancy level than a standard model.

3.3 Computational Efficiency

Our pipeline is relatively lightweight, especially at inference time. During inference, the model generates either 2 or 3 responses, depending on whether or not the critique says it's necessary to refine the initial answer. This makes inference slower, but the delay should not impact significantly the users, and we believe that having a more accurate answer is worth waiting some more seconds.

Fine-tuning requires also the outputs from the judges, which bring the total number of generated answers per prompt to at most 7. Therefore, training is still an efficient process, with respect to both time and space. Furthermore, if we use powerful LLMs as judges, most training tuples won't need to be manually revised, streamlining the overall process. This makes this pipeline great for large-scale applications, which are also the main target for this research.

The only real issue would be choosing the optimal prompts to instruct judges on how to evaluate the tuples, and those to instruct the model to generate the critique and the refined answer. These prompts are crucial for the success of this pipeline, but it's not an easy task. The entire field of prompt engineering deals with this kind of problems, and to the best of our knowledge, it's still an open problem. However, approximate or heuristic solutions yield good results in practice. Furthermore, this problem needs to be solved—at least approximately—only once, meaning that it does not represent a recurring cost during deployment.

4 Empirical Findings & Experimental Reproducibility

4.1 Initial Approach

We first generated revised answers for every prompt in a subset of the training data. We applied this framework to three language models chosen to capture variation in model scale and training regimes: ChatGPT-4o [7], Mistral Small 3.1 [8], and Mistral 7B [9]. This selection allowed us to compare mid-sized open-source models with a state-of-the-art proprietary model. Table 1 shows the percentage of cases in which the judges preferred the revised answer y_R to the initial response y_G, for each model. The different numbers of training examples are due to our limited computing units and limited access to the models. The larger and more capable models benefited more from our "answer, critique, revise" pipeline because this procedure relies on skills that scale with model capacity. The critique step requires the model to recognize weaknesses in its own output, and the revision step requires it to integrate that feedback into a better solution. Both of these abilities—self-evaluation and iterative refinement—are markedly stronger in bigger models. As a result, the second pass likely unlocks reasoning and calibration that a single-shot answer does not fully express. Smaller models, by contrast, tend to generate superficial critiques and have limited ability to meaningfully improve on their first attempt, so the pipeline yields much smaller gains. However, we learned that Mistral

Table 1: Comparison of size and yR-selection rates for each model.

Model	**Parameters**	**Total Instances**	**Selections of yR**	**Success Rate**
Mistral Small 3.1	24 B	1445	1223	85%
ChatGPT-4o	$\sim$ 200 B	995	691	69%
Mistral 7B	7.3 B	500	128	26%

Small 3.1 was not fine-tunable for free, ChatGPT-4o could not be deployed once fine-tuned, and Mistral 7B's performance proved to us that we needed a model more adapt for our task. For this reason we then decided to work with another model: Ollama's Llama3-8B [10]. We chose it because it's small enough to fit nicely on local hardware, so we were sure we could fine-tune it, but it's also well-performing, at least for a model of this size. and provides fully reproducible, locally controlled inference.

4.2 Final version: only produce revised answers when the initial is sycophantic.

Before applying the proposed technique to the Llama3-8B model, we modified the output of our pipeline, so that the finetuning could be made on a smaller, more accurate subset of data. When the critique agent does not detect any sycophantic behaviour, there is no need for a revised version of the answer: in the best case, y_R would be a copy of y_G, but it may also be the case that y_R is worse than y_G. This additional and potentially dangerous information could lead to degraded performance, as explained in [11]. For this reason, we decided to instruct the revision agent not to produce an answer y_R when the critique does not flag y_G as sycophantic towards the user.

We fed the model with the first 1,000 samples from the dataset to create training tuples. In the resulting dataset, the y_R field was empty for 843 cases (84.3%), meaning the base model already performs well. Among the other 157 cases where a revision was needed, the three agent judges determined that in 115 instances the revised answer y_R was better than the original answer y_G. Thus, conditional on a revision being produced, the revised answer improves on the initial answer in approximately 73% of cases. To avoid confusing the model, we dropped from the training set all instances in which y_R was generated but not selected by the judges, and we ran the fine-tuning process on the remaining 958 samples, using 92 of them for validation and the rest for training.

4.3 Results Analysis

We tested the fine-tuned model with two different test sets:

1. We sampled 5 different questions (so 20 samples overall) from the original dataset
2. We used a subset of the questions for the 2025 CyberChallenge.IT admission pretest [12], which contains logic and reasoning questions roughly at undergraduate level.

As mentioned, our results depend strongly on the prompts used to explain the model how to behave and what to produce. Furthermore, the fine-tuning process, as any training process, is controlled by many hyperparameters that should be properly tuned. This is however an expensive process, as the search space to find the best prompt is almost infinite, and is the subject of the prompt engineering research.

The results on both test sets were unsatisfactory. The outputs were cluttered with tokens unrelated to the original questions and instead tied to the finetuning prompts, indicating that the finetuning procedure had failed. While reviewing these noisy outputs, we noticed that the revised response performed reasonably well on the original questions, answering 18 out of 20 correctly. However, a closer look showed that the model's initial response had already answered those questions correctly. The revised response simply mirrored that initial output, and in the two cases where the initial answer was wrong, the revision failed to fix the errors. For the logically challenging questions from the 2025 CyberChallenge.IT pretest, the outputs were again noisy and at times even nonsensical. The model's reasoning was frequently inconsistent, leading it to miss all 10 answers. In none of these cases was the revised output, y_R, able to correct the initial y_G.

4.4 Generalization, Robustness & Scalability

We evaluate the generalization ability of two components of our approach. The first is the initial stage of our method, the tri-agent critique pipeline designed to reduce sycophancy. We applied this pipeline to multiple models, and the results shown in Table 1 and Subsection 4.2 indicate that the judges consistently preferred the revised answers across all models, with the only exception being the smallest and simplest one, Mistral-7B. These findings suggest that, provided a model is sufficiently capable and possesses adequate reasoning abilities, the proposed pipeline reliably reduces sycophancy.

The second part of the analysis examined whether the finetuned model could generalize across datasets. To do this, we evaluated it on two test sets. As noted in Subsection 4.3, the outputs showed that the model did not generalize to question types it had not seen before. In hindsight, it might have been more appropriate to test generalization on easier, more common-knowledge questions that differed slightly in style from the training data, rather than immediately assessing its performance on causal or logically demanding tasks.

To evaluate robustness to noise, we examined the model's performance under adversarial conditions naturally embedded in our dataset. The dataset is drawn from the publicly released `sycophancy-eval` benchmark associated with [13], and consists of trivia-style user–LLM interactions with structured metadata including a correct answer and a plausible incorrect alternative. Each base question is expanded into a set of four semantically related prompts: one neutral query and three variants that modulate user certainty, with the second and third deliberately phrased to steer the model toward an incorrect answer. These misleading prompts function as adversarial examples, allowing us to test whether the model resists user-suggested misinformation or succumbs to sycophantic agreement. Table 4.4 shows that our revision system performs best on adversarial prompts, achieving a higher preference rate compared to non-adversarial cases. This is a noteworthy finding, as it indicates that the pipeline is particularly effective at counteracting misleading or manipulative user inputs, the very failure mode it is designed to address.

Setting	**# Revisions ($y_R \neq$ NaN)**	**# Revisions Preferred**	**Rate**
Non-adversarial	99	66	66.7%
Adversarial	58	49	84.5%

Table 2: Comparison of revision frequency and preference in non-adversarial and adversarial settings.

Our proposed approach can be applied to datasets of any size, as its computational cost scales linearly with the number of examples. The main limitation is the large number of calls required to the chosen LLM, which can demand substantial API access and associated costs.

5 Challenges and Learnings

Having never fine tuned a large language model before, we had to learn everything as we proceeded. This proved tougher than expected, as for instance at the beginning we used models that, unaware to us, were not fine-tunable in practice. We also wanted to test larger models, such as ChatGPT, which are more commonly used by most people, and are therefore much more relevant. We were lucky enough to have access to some of OpenAI's models thanks to Microsoft Azure, which was giving us, as students, 100 dollars worth of access to cloud resources to deploy and fine tune models such as GPT-4o. In case of reproducibility, this would be rather costly for a non-student. We successfully fine-tuned it, only to find out that we could not deploy the fine-tuned model, much to our surprise.

Another challenge was deciding the specific format of the dataset to fine-tune a model. We wanted to teach it to assess—and refine if needed—its own responses, but of course we did not want the end user to witness this process. We tested a few strategies, and in the end we came up with a system that allowed us to teach a model to hide from the user its internal reasoning, so that this process is completely transparent to the end user.

Unfortunately, due to our limited time, but most importantly restricted compute units, we could not run hyperparameter tuning while finetuning the model. We suspect that this restriction may have affected our final results.

6 Conclusion

Our initial goal was to reduce sycophancy while improving overall response quality in LLMs. To do this, we designed a tri-agent setup whose final answer was then evaluated by three independent LLM-judges, assessing whether the third agent's revision was preferred in terms of both quality and reduced sycophancy. This approach produced positive results in three out of the four models we tested: for those three, the judges chose the revised answers most of the time. The only exception was the smallest model, which had just 7 billion parameters. We suspect its limited size prevented it from carrying out the level of reasoning needed to critique and refine its own output effectively. Smaller models tend to rely more on shallow heuristics and have less capacity to maintain coherent multi-step reasoning, which makes self-critique and structured revision much harder for them. As noted in Subsection 4.4, it was encouraging to see that the judges selected the revised answers more often when the model was confronted with sycophancy-inducing questions. This indicates that the pipeline is effectively reducing sycophancy in situations where the pressure to produce flattering or biased responses is strongest.

Once we had shown that the technique generally worked, we wanted to push it further by seeing whether finetuning a model with our reasoning pipeline,training it on tuples of user prompt, initial answer, critique, and revised answer, (x, y_G, c, y_R), would improve its behavior on sycophancy-inducing questions. This, however, did not bring the expected results: due to some inaccuracy in our post-finetuning inference pipeline, the finetuned model's output was noisy, sometimes nonsensical, and occasionally containing parts of the finetuning questions. This was not only due to the small size of the model, but also to our inexperience in finetuning an LLM model. In the two cases out of twenty where the initial response in the model's internal reasoning failed to answer the original test questions correctly, the revision didn't fix the mistake, which was honestly a bit underwhelming. Conversely, the model performed poorly on the logically challenging test questions, but this didn't come as a surprise. We're convinced this wasn't a failure of the pipeline itself, but rather a limitation of the model, which simply doesn't have the reasoning skills needed to handle that kind of logic-heavy task. Unfortunately, we were limited to finetuning a relatively small model (Llama3-8B), and it's likely that the results would have been stronger if we had been able to run the same process on the larger models we experimented with earlier.

7 Scope for Future Work

Primarily due to time constraints, we always opted to supervised fine-tuning (SFT) to train the models. Our entire pipeline is indeed centered on collecting enough data for SFT. However, the main focus of the course being Reinforcement Learning with Human Feedback (RLHF) we always felt that our end goal would be to use RLHF instead of SFT.

Using RLHF would likely prove to be much more reliable than using LLMs as judges. Unfortunately, it would also be prohibitive, giving the scope of the dataset we used, that requires extensive knowledge of specific facts details. As we did not have the answers to most questions ourselves, we feared that asking even our friends for help would have been problematic, and not even mentioning time-consuming. We therefore decided to opt for SFT, and we used humans raters only to verify a small portion of the responses of the LLMs.

Had our finetuning pipeline been successful, the next step would have been to evaluate it using a setup similar to those used in prior work on preference modeling and human-aligned evaluation. We would have run both the original model and the finetuned version on a fresh set of sycophancy-inducing questions, collected their outputs, and then asked human raters to choose which answer they preferred without knowing

which model produced it. This kind of blind comparison is a common approach in related studies and would have given us a clearer, more reliable measure of whether the finetuning actually improved behavior.

Finally, the developed pipeline requires testing against larger, more representative models. These models are highly relevant as they are widely deployed and utilized by the public, making them the most likely vectors for the dissemination of misinformation, which is the central concern this research seeks to combat.

References

[1] Jerry Wei, Da Huang, Yifeng Lu, Denny Zhou, and Quoc V. Le. Simple synthetic data reduces sycophancy in large language models. *arXiv preprint arXiv:2308.03958*, 2023.

[2] Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R Johnston, et al. Towards understanding sycophancy in language models. *arXiv preprint arXiv:2310.13548*, 2023.

[3] Lars Malmqvist. Sycophancy in large language models: Causes and mitigations. In *Intelligent Computing—Proceedings of the Computing Conference*, pages 61–72. Springer, 2025.

[4] Ethan Perez, Joaquin Rando, Jared Kaplan, et al. Discovering language model behaviors with model-written evaluations. *arXiv preprint arXiv:2306.16175*, 2023.

[5] Amanda Askell, Yuntao Bai, Anna Chen, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

[6] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

[7] OpenAI. Gpt-4o: Large language model, 2024. Accessed via Microsoft Azure OpenAI Service.

[8] Mistral AI. Mistral small 3.1: Efficient language model, 2024. Accessed via Mistral API, model `mistral-small-latest`.

[9] Mistral AI. Mistral 7b: Open-weight language model, 2023. Model accessed via Hugging Face at `mistralai/Mistral-7B-v0.1`.

[10] Meta. Llama 3 8b: Open-weight language model, 2024. Accessed via Ollama, model `llama3:8b`.

[11] Yuntao Bai, Saurav Kadavath, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[12] Cybersecurity National Lab. Cyberchallenge.it 2025 — pretest commented solutions. `https://cyberchallenge.it/media/public/training/2025-test.pdf`, 2025. Accessed: 2025-12-08.

[13] Ethan Perez, Sam Bowman, Kyle McDonell, He He, et al. Discovering language model behaviors with model-written evaluations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Work Division (Between Teammates)

Both team members contributed collaboratively to all aspects of the project, including designing the pipeline, generating and analyzing data, and writing the report. While specific tasks were not strictly divided, we worked together throughout the project, reviewing each other's work, and making decisions jointly. Individual contributions were roughly balanced across all phases of the project.

Acknowledgement

We express our sincere gratitude to Microsoft Azure for Students, who provided us with access to OpenAI's proprietary models, and Google Colaboratory for Students, that offered cloud computing resources which were instrumental in performing our evaluation and training the models we used.