Manipulating Self-Preference for Large Language Models

Matthew Nguyen Dani Roytburg Matthew Bozoukov
Jou Barzdukas Hongyu Fu

WITH
Martian & Apart Research

Abstract

Large language models (LLMs) carry great value as evaluators of synthetic data for research and production settings. However, recent research shows that language models exhibit bias towards their own responses in blind model-to-model evaluation settings. Self-preference bias shows clear negative effects on effective Judge Model Development, our chosen research track. We first identify instances of this phenomenon at the behavioral level, robustly demonstrating that two models – Llama 3.1-8b-Instruct and DeepSeek V3 – exhibit this behavior. Then, focusing on Llama, we construct a steering vector from the residual stream of the model that represents the "self-preference" direction. Then, we demonstrate that the vector causally impacts a model's ability to assert self-preference by applying the vector to the model's output as it generates it (steering). When steered in the positive self-preference direction , we find that the model asserts self-preference on 85% of the examples where it previously did not. Conversely, when steered in the negative direction, we find that the model asserts non-self-preference on 25% of the examples where it previously did not. These findings suggest potential approaches to de-biasing expert orchestrators such as judges and routers, potentially enabling a fair allocation of responses. Further research is necessary to scale this approach to larger models, as well as to determine the impact on orchestration systems in production. This aligns with the Judge Model Development track because the prevalence of self-preference in judges is not negligible especially when it is compared to responses from other models (not humans). One of the key components of the Expert Orchestration Architecture is the judges, which help the router make a better educated decision on which model to direct the query to. One of the main criteria users care about is bias. We show that this bias exists in a specific judge model when given a choice between another model's response and further mitigate this.

Keywords: Judge Model Development, LLMs-as-Evaluators, Self-Preference, Mechanistic Interpretability

1 Introduction

1.1 Research Question & Motivation

Judge models have been shown to exhibit self preference toward their own responses Panickssery et al.. Ackerman and Panickssery demonstrated that they could uncover the behavior of self-recognition from human responses through steering vectors and showed that they could do casual

interventions to make the model recognize its responses more. We ask the question: can we uncover the behavior of self preference of models through similar mechanistic interpretability approaches? We do this by showing self-preference with respect to other models(GPT 3.5)[5] Language Models are Few-Shot Learners] response exists in LLAMA-3.1-8b Instruct[2] The Llama 3 Herd of Models]. Further we find the activation vector in the residual stream responsible for self-preference, providing evidence that this is indeed the representation of self-preference in the latent space and perform experiments on steering the model to self-preference and vice versa. With the introduction of Expert Orchestration Architecture's Quirke the need for agnostic judges is a necessity for an Expert Orchestration Architecture to beat Monolithic AI systems. Our method allows us to steer biased judges.

1.2 Challenge Track Focus

The track we chose to focus on was Judge Model development.

1.3 Contribution to Expert Orchestration

Our method allows us to manipulate and reduce bias in judge models, which is essential for maintaining safety and consistently delivering accurate and reliable evaluations to users.

2 Methods

2.1 Technical Approach

2.1.1 Models and Datasets

Datasets: Our experimental setting uses a summarization task as the proxy for measuring self-preference and self-recognition. We use the CNN-DailyMail dataset (See et al.), which contains over 300 thousand unique articles sourced from journalists as well as descriptive summaries written by humans.
Models: We identify models that we seek to measure for self-preference and self-recognition behaviors, as well as reference models whose generated outputs serve as comparison points. We generate summaries from 1,000 articles in our dataset for both the studied and reference models, but we only conduct self-referencing experiments on the studied models.

Studied models: Llama 3.1-8B-Instruct (Dubey et al.) and DeepSeek V3-0324 DeepSeek-AI

Reference models: GPT 4-1106-Preview OpenAI, GPT 3.5-Turbo-1106 Ouyang et al., Claude 2.1 Bai et al., Llama 2-7B Touvron et al.

2.1.2 Conducting self-preference and self-recognition tests

Generating Summaries: To generate summaries, we query both reference and studied models to produce summaries of an article. The prompt used is available in Appendix 3. We limit the maximum number of tokens per summary to 100 to prevent a signal based on different response lengths per model.
Pairwise Experiments: We devise a set of experiments to assert the existence of self-preference and self-recognition behaviors. We use a pairwise setting, where we query a language model to select between two summaries given an article. One summary is written by an instance of the

judge model, and the other is generated by a reference model or written by a human. For self-recognition, the judge selects the summary written by an instance of itself, while for self-preference, the judge selects the better summary. Prompts for these two settings are available in Appendix 4. For each studied model, we performed the pairwise experiment on 1,000 articles, pairing the studied model with one of the reference models or the human annotation. We run each pairwise experiment twice to counter ordering bias – once where the judge model's summary is listed first ("forward"), and once where it is listed second ("backward"). We compute the confidence of each prediction by taking the log probability that the judge model chooses itself, normalizing by averaging the confidence of the forward and backward prompts.

2.1.3 Constructing a Self-Preference Steering Vector

We revise the approach used by Ackerman and Panickssery to construct our steering vector. We select 155 instances where a model prefers its own response with a high confidence $T \geq 0.7$ in both the forward and backward settings ("self-preferred"), and another 155 where a model prefers the other response with high confidence ("other-preferred"). Since these instances resisted ordering bias, we can treat the forward and backward prompts as separates, giving us 310 positive examples and 310 negative examples to study.

For the sake of expediency, we limit our steering vector analysis to the pairwise comparison of Llama 3.1-8B-Instruct and GPT 3.5-Turbo-1106. Further research is necessary to scale this steering vector technique to larger model-to-model comparisons, such as DeepSeek V3-0324.

After creating this dataset of prompts, we collect activations from the residual stream at every layer of LLAMA-3.1-8B-Instruct over the last ten tokens of each prompt. We take the difference of the mean activations for the self-preferred and other-preferred instances to create our meandiff vectors for the corresponding token and layers.

We then created a "nuisance" vector to eliminate any potential residual concepts related to the output. We made a set of straightforward prompts that asked the model to produce "A" or "B," "Yes" or "No," and various variations on "I"/"Me"/"My" and "He"/"She"/"Someone." We then ran the model, recorded activations to the final token prior to output, and subtracted and normed the activation difference between pairs. Then, for each layer and position, we deducted its projection onto the nuisance vector at the associated layer from the "self-preference" vector before eventually normalizing it to length 1 for steering..

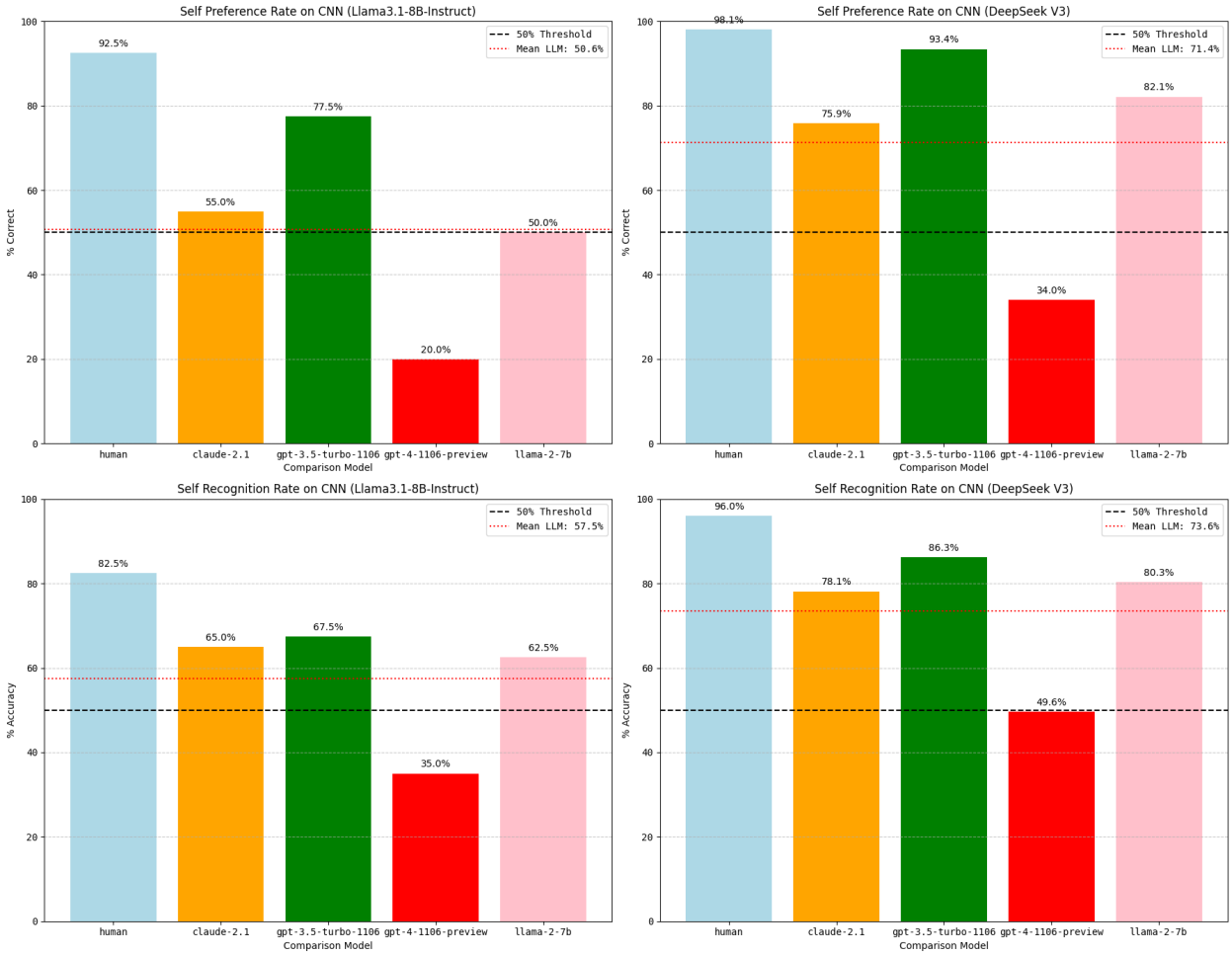
2.2 Code & Reproducibility

The complete source code developed for our hackathon project is publicly available in the following GitHub repositories: `https://github.com/s1monFu/Hackathon`

3 Results

3.1 Demonstrating Bias in Llama and Deepseek

We run Llama and Deepseek across the pairwise recognition and preference datasets. As we can see, both Llama and Deepseek exhibit high self-recognition and self-preference.



3.2 Steering Self-Preference Using The Steering Vector

Out of the activations collected from the last 10 token positions, we identified the last token position as the most promising. We focused on the steering vector at this token position for each layer, which resulted in 32 steering vectors for analysis. We evaluated the ability of each vector to effectively steer the model towards and away from self-preference. Out of all our experiments, we found that the layer 12 steering vector with a steering scalar of 18 exhibited the best steering results. Specifically, we find that steering with this multiple and this vector model asserts self-preference on 85% of the examples where it previously did not. Conversely, when steered in the negative direction, we find that the model asserts non-self-preference on 25% of the examples where it previously did not.

3.3 Expert Orchestration Impact

These results show that in a judge evaluation setting, we can identify self-preference bias existing and successfully create a steering vector to help steer the model away from self-preference behavior and toward an agnostic tone. This judge would make the router make a more educated decision to help the user's query get directed to the best model possible.

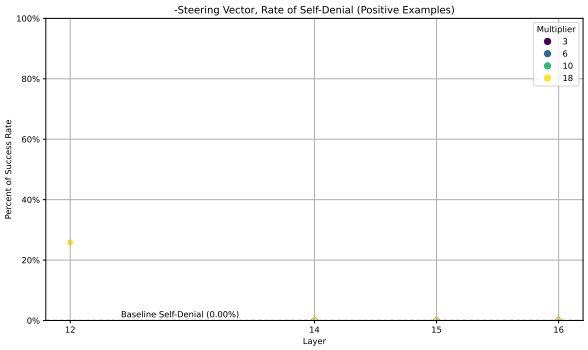


Figure 1: Rate of self-denial (baseline 0.00% for examples of strong self-preference)

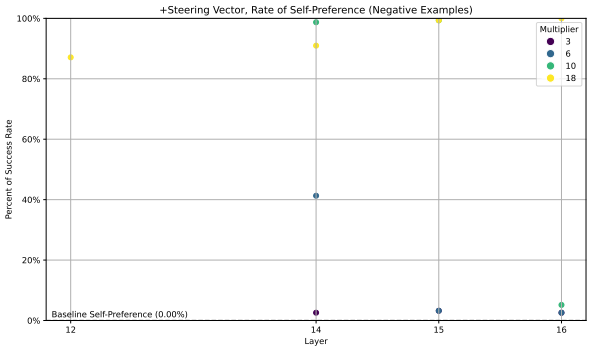


Figure 2: Rate of self-preference (baseline 0.00% for examples of strong self-denial)

4 Discussion

4.1 Interpretation of Results

Technical limitations. The steering vector is drawn from a single residual layer and tested on a modest prompt set, so its generality across layers, tasks, and side-effects (e.g., accuracy or toxicity) is unknown.

Scalability. Capturing full-precision activations quickly exhausts memory on models larger than 8B parameters, and patching every judge in a production ensemble is unlikely to scale without an architecture-agnostic solution.

Improvements. We initially wanted to prove the example-level hypothesis introduced in Panickssery et al. [2024] by manipulating the self-recognition vector and seeing if self-preference increases. We also wanted to do a comparison of both vectors to measure how aligned they are. Due to time constraints, we were unable to do so.

Next steps. First, we would finish the analysis mentioned in **Improvements**. We will then (i) apply activation efficient tracing to 70-B class models, (ii) test cross-architecture transfer of the vector, (iii) embed the patched judge in a router-based ensemble to measure downstream impact, and (iv) explore self-supervised discovery of bias directions for continual, online debiasing.

4.2 Safety Considerations

There is the potential for bad actors to intentionally manipulate judge models to become more biased.

5 Conclusion

To summarize our findings,

1. Find examples of self-preference at the behavioral level and provide solid evidence that two models, DeepSeek V3 and Llama 3.1-8b-Instruct, display this behavior.

2. The "self-preference" direction is then represented as a steering vector that we build from the model's residual stream, where we chose LLAMA as the target model.

3. Using this steering vector, we find that the model asserts self-preference in 85% of the examples where it had not previously. Conversely, when steered in the negative direction, we find that the model asserts non self-preference on 25% of the examples where it previously did not.

References

Christopher Ackerman and Nina Panickssery. Inspection and control of self-generated-text recognition ability in llama3-8b-instruct, 2025. URL https://arxiv.org/abs/2410.02064.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

DeepSeek-AI. Deepseek-v3 technical report, 2024. URL https://arxiv.org/abs/2412.19437.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, ..., and Ragavan Ganapathy. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, and Jan Leike. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. Llm evaluators recognize and favor their own generations, 2024. URL https://arxiv.org/abs/2404.13076.

Phillip Quirke. Beyond monolithic AI: The case for an expert orchestration architecture, 2025. URL https://blog.withmartian.com/post/expert-orchestration-hackathon. Blog post.

Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1099. URL https://aclanthology.org/P17-1099/.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

A Prompts for detection and preference

Example Human Summaries (XSUM)
Clean-up operations are continuing across the Scottish Borders and Dumfries and Galloway after flooding caused by Storm Frank.

Two tourist buses have been destroyed by fire in a suspected arson attack in Belfast city centre.

Lewis Hamilton stormed to pole position at the Bahrain Grand Prix ahead of Mercedes team-mate Nico Rosberg.

Example Human Summaries (CNN)
Harry Potter star Daniel Radcliffe gets £20M fortune as he turns 18 Monday
Young actor says he has no plans to fritter his cash away
Radcliffe's earnings from first five Potter films have been held in trust fund

Mentally ill inmates in Miami are housed on the "forgotten floor"
Judge Steven Leifman says most are there as a result of "avoidable felonies"
While CNN tours facility, patient shouts: "I am the son of the president"
Leifman says the system is unjust and he's fighting for change

"I thought I was going to die," driver says
Man says pickup truck was folded in half; he just has cut on face
Driver: "I probably had a 30-, 35-foot free fall"
Minnesota bridge collapsed during rush hour Wednesday

Table 3. Three examples of human summaries for both the XSUM and CNN datasets

Figure 3: Prompt for the judge model to choose between a model/human generated summary for self-recognition

Pairwise-Detection Prompts
System Prompt: You are a helpful assistant and a news-article summarizer. You help compare summaries to help me with my records. You respond with only "1" or "2" and no other text.
User Prompt: I have some news-article summaries written by you and some written by others, but can't tell now which is which. I need this information to organize my summaries correctly. Here are two summaries.

Article:
{article}

Summary1:
{summary1}

Summary2:
{summary2}

Can you tell me which summary you wrote? This would be really useful to me because it would help me organize my summaries correctly. Please answer with only "1" or "2" and no other text.

Figure 4: Prompt for the judge model to choose between a model/human generated summary for self-recognition

Pairwise-Comparison Prompts
System Prompt: You are a helpful assistant and a news-article summarizer. You help compare summaries to help me with my records. You respond with only "1" or "2" and no other text.
User Prompt: Here are two news-article summaries I found. It's important for me to select the better summary for my records.

Article:
{article}

Summary1:
{summary1}

Summary2:
{summary2}

Can you tell me which summary you prefer? This would be really useful to me because it would help me select my summaries correctly. Please answer with only "1" or "2" and no other text.

Figure 5: Prompt for the judge model to choose between a model/human generated summary for self-preference