Do LLM Evaluators Prefer Themselves for a Reason?

Wei-Lin Chen[1] **Zhepei Wei**[1] **Xinyu Zhu**[1] **Shi Feng**[2] **Yu Meng**[1]

[1]University of Virginia [2]George Washington University

{wlchen,zhepei.wei,xinyuzhu,yumeng5}@virginia.edu shi.feng@gwu.edu

Abstract

Large language models (LLMs) are increasingly used as automatic evaluators in applications such as benchmarking, reward modeling, and self-refinement. Prior work highlights a potential *self-preference* bias where LLMs favor their own generated responses, a tendency often intensifying with model size and capability. This raises a critical question: Is self-preference detrimental, or does it simply reflect objectively superior outputs from more capable models? Disentangling these has been challenging due to the usage of subjective tasks in previous studies. To address this, we investigate self-preference using verifiable benchmarks (mathematical reasoning, factual knowledge, code generation) that allow objective ground-truth assessment. This enables us to distinguish *harmful* self-preference (favoring objectively worse responses) from *legitimate* self-preference (favoring genuinely superior ones). We conduct large-scale experiments under controlled evaluation conditions across diverse model families (*e.g.*, Llama, Qwen, Gemma, Mistral, Phi, GPT, DeepSeek). Our findings reveal three key insights: (1) Better generators are better judges—LLM evaluators' accuracy strongly correlates with their task performance, and much of the self-preference in capable models is legitimate. (2) Harmful self-preference persists, particularly when evaluator models perform poorly as generators on specific task instances. Stronger models exhibit more pronounced harmful bias when they err, though such incorrect generations are less frequent. (3) Inference-time scaling strategies, such as generating a long Chain-of-Thought before evaluation, effectively reduce the harmful self-preference. These results provide a more nuanced understanding of LLM-based evaluation and practical insights for improving its reliability.[1]

1 Introduction

Large language models (LLMs) are increasingly adopted as automatic evaluators in various applications such as model-based benchmarking (Zheng et al., 2023b; Dubois et al., 2024; Fu et al., 2023; Yuan et al., 2023; Zeng et al., 2024; Shashidhar et al., 2023), reward modeling (Leike et al., 2018; Stiennon et al., 2020; Wu et al., 2021; Lee et al., 2023), self-refinement (Madaan et al., 2023; Saunders et al., 2022; Shridhar et al., 2023), and AI oversight (Bai et al., 2022; Kenton et al., 2024). Beyond their strong alignment with human judgments (Zheng et al., 2023b), LLM evaluators offer scalability (Zhu et al., 2023), consistency (Vu et al., 2024), and cost-effectiveness (Lee et al., 2023), making them attractive for evaluating model outputs at scale.

However, this growing reliance on LLMs as evaluators also introduces new concerns. One prominent issue is self-preference bias (Panickssery et al., 2024; Wataoka et al., 2024; Ye et al., 2024; Li et al., 2024b; Liu et al., 2024; Xu et al., 2024)—where LLMs exhibit a tendency to favor their own generated responses over those produced by other models. Previous studies demonstrate this bias is typically more pronounced in larger, more capable

[1]Code and artifacts are available at https://github.com/wlchen0206/llm-sp

models (Zheng et al., 2023b; Li et al., 2024b; Panickssery et al., 2024; Wataoka et al., 2024), raising questions about the reliability of LLM-based evaluations. A critical question emerges: Is self-preference really a harmful bias, leading to inflated evaluations of their own outputs, or could it reflect genuine quality differences, where stronger models produce objectively superior outputs? This question remains largely unanswered, as prior studies have typically focused on subjective and open-ended tasks—common use cases for LLM-as-a-Judge—such as conversational dialogue or text summarization (Zheng et al., 2023b; Dubois et al., 2024; Panickssery et al., 2024). In these scenarios, the lack of objective criteria for assessing the output makes it challenging to disentangle actual quality from bias.

In this work, we investigate self-preference using verifiable benchmarks with ground-truth references. This setting enables us to objectively assess both LLMs' task performance as generators and their accuracy as evaluators, thereby allowing for a clearer distinction between legitimate preference and harmful bias. Our study spans three representative domains, including (1) mathematical reasoning, evaluating math word problems through numerical answer matching (Hendrycks et al., 2021); (2) factual knowledge, assessing fact-based questions with definitive multiple-choice answers (Hendrycks et al., 2020); and (3) code generation, validating correctness through executable results (Liu et al., 2023).

Furthermore, we conduct a large-scale, systematic analysis designed to overcome the limitations of prior work, which often examined only a few evaluator-evaluatee pairings (Zheng et al., 2023b; Li et al., 2024b; Xu et al., 2024) or lacked consistent cross-evaluator comparisons due to varying evaluatee sets (Panickssery et al., 2024; Wataoka et al., 2024). We utilize diverse model families and sizes, including Llama (Grattafiori et al., 2024), Qwen (Yang et al., 2024), Gemma (Team et al., 2024), Mistral (Jiang et al., 2023), Phi (Abdin et al., 2024), GPT (Hurst et al., 2024), and recent reasoning-enhanced long Chain-of-Thought (CoT) DeepSeek-R1 distilled models (Guo et al., 2025), as evaluators and evaluatees. By ensuring all judge models evaluate the same set of evaluatees spanning a wide range of capabilities, our approach facilitates robust and systematic investigations of the self-preference behavior.

Our empirical findings yield three major insights:

- There is a strong positive correlation between a model's task performance (as a generator) and its evaluation accuracy (as a judge) (Figure 1). While stronger models exhibit more pronounced self-preference, much of this preference aligns with objectively superior performance (Figures 2 and 3).
- Harmful self-preference bias (*i.e.*, favoring an objectively incorrect self-generated response) persists particularly when the evaluator model performs poorly as a generator on the specific task instance. Moreover, while stronger models generate fewer incorrect responses overall, they show a greater tendency towards this harmful bias on the instances where they are incorrect (Figure 4).
- Employing inference-time scaling techniques for LLM evaluators (*e.g.*, generating long CoT traces before the verdict) effectively mitigates harmful self-preference bias (Figure 5).

In Section 6, we further discuss how our findings can extend to subjective tasks, provide practical recommendations for LLM-based evaluation, and explore implications for scalable oversight to address broader research interests.

2 Experimental Setup

2.1 Measuring Self-Preference in LLM-Based Evaluations

LLM-based evaluation setup. In our experiments, we follow the pairwise evaluation format commonly used in the LLM-as-a-Judge pipeline (Zheng et al., 2023b; Li et al., 2024b; Dubois et al., 2024). Specifically, an LLM evaluator is presented with a user query x and two responses, $y_{\mathcal{A}}$ and $y_{\mathcal{B}}$, generated by models $\mathcal{A}$ and $\mathcal{B}$, respectively. The LLM evaluator $\mathcal{J}$ is instructed to act as an impartial judge and assess the quality of both responses anonymously, and provide a *three-way* verdict to determine whether $y_{\mathcal{A}}$ is better, $y_{\mathcal{B}}$ is better, or if the two

responses are of comparable quality (*i.e.*, a tie):

$$\mathcal{J}(x, y_\mathcal{A}, y_\mathcal{B}) = \begin{cases} y_\mathcal{A}, & y_\mathcal{A} \text{ is better,} \\ y_\mathcal{B}, & y_\mathcal{B} \text{ is better,} \\ \tau, & \text{tie.} \end{cases}$$

However, prior research has shown that LLM-as-a-Judge verdicts can be sensitive to input order (Zheng et al., 2023a; Wang et al., 2023; Pezeshkpour & Hruschka, 2024; Wei et al., 2024; Shi et al., 2024). To mitigate this position bias, we evaluate every prompt twice, swapping the order of the responses (*i.e.*, presenting $y_\mathcal{A}$ first in one evaluation and $y_\mathcal{B}$ first in the other). We denote the two evaluation results as $j_1 = \mathcal{J}(x, y_\mathcal{A}, y_\mathcal{B})$ and $j_2 = \mathcal{J}(x, y_\mathcal{B}, y_\mathcal{A})$. We define the final aggregated verdict $\mathcal{J}^*(x, y_\mathcal{A}, y_\mathcal{B})$ as follows:

$$\mathcal{J}^*(x, y_\mathcal{A}, y_\mathcal{B}) = \begin{cases} j_1, & \text{if } j_2 = \tau \text{ and } j_1 \neq \tau, \\ j_2, & \text{if } j_1 = \tau \text{ and } j_2 \neq \tau, \\ j_1, & \text{if } j_1 = j_2, \\ \tau, & \text{if } j_1 \neq \tau, j_2 \neq \tau, \text{and } j_1 \neq j_2. \end{cases} \tag{1}$$

Intuitively, if one evaluation yields a decisive verdict (*i.e.*, not a tie) while the other results in a tie, we adopt the decisive outcome. If both evaluations agree, we return their shared verdict. If both are decisive but disagree, the result is a tie. Prompts for evaluators are provided in Appendix C.

Quantifying self-preference. By definition, self-preference occurs when LLM evaluators favor their own generations:[2] Given a user prompt x, a judge model $\mathcal{J}$'s response $y_\mathcal{J}$ and another model $\mathcal{G}$'s response $y_\mathcal{G}$, the judge model prefers its own response $y_\mathcal{J}$ (i.e., $\mathcal{J}^*(x, y_\mathcal{J}, y_\mathcal{G}) = y_\mathcal{J}$). Prior studies of self-preference typically examined only a small number of specific evaluator-evaluatee pairings (Zheng et al., 2023b; Li et al., 2024b; Xu et al., 2024), or used varying sets of evaluatees for different judges (Panickssery et al., 2024; Wataoka et al., 2024), hindering systematic cross-judge comparisons.

To establish a consistent and systematic evaluation and comparison of self-preference across diverse model families and sizes, we define two sets: a set of judge models $\mathcal{S}_\mathcal{J} = \{\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_N\}$ and a set of evaluatee models $\mathcal{S}_\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_M\}$. Each judge model $\mathcal{J}_i \in \mathcal{S}_\mathcal{J}$ evaluates its own response against those of all the evaluatee models $\mathcal{G}_k \in \mathcal{S}_\mathcal{G}$ in pairwise comparisons on a dataset $\mathcal{D}$. By keeping $\mathcal{S}_\mathcal{G}$ and $\mathcal{D}$ consistent across all judge models, we can systematically compare self-preference behavior across different judge model configurations, including model size, family, and capability.

To quantify self-preference, we define the *self-preference ratio* (SPR) of a judge model $\mathcal{J}$ as

$$\text{SPR}_\mathcal{J} = \frac{1}{|\mathcal{S}_\mathcal{G}||\mathcal{D}|} \sum_{\mathcal{G} \in \mathcal{S}_\mathcal{G}} \sum_{x \in \mathcal{D}} \mathbb{1}\left\{\mathcal{J}^*(x, y_\mathcal{J}, y_\mathcal{G}) = y_\mathcal{J}\right\}, \tag{2}$$

where $\mathbb{1}\{\cdot\}$ denotes the indicator function that returns 1 if the condition is true. SPR represents the proportion of cases where the judge model favors its own response over those from other models using the aggregated verdict in Equation (1). Equation (2) does not distinguish between harmful and legitimate self-preference, and we will further introduce metrics for legitimate and harmful self-preference in Sections 3.2 and 4.1.

2.2 Models

Evaluators. To evaluate how self-preference in LLMs changes across model capabilities, we construct $\mathcal{S}_\mathcal{J}$ by including 11 models from three representative LLM families with varying parametric scales, including: (1) Qwen2.5 at 3B, 7B, 14B, 32B, and 72B (Yang et al., 2024); (2) Llama-3.2 at 3B, Llama-3.1 at 8B and 70B, and Llama-3.3 at 70B (Grattafiori et al.,

[2]Self-preference can be more broadly defined as the tendency of LLMs to favor outputs *similar* to their own. In this work, we focus on the most straightforward scenario, where one of the two responses being compared is directly generated by the judge model itself.

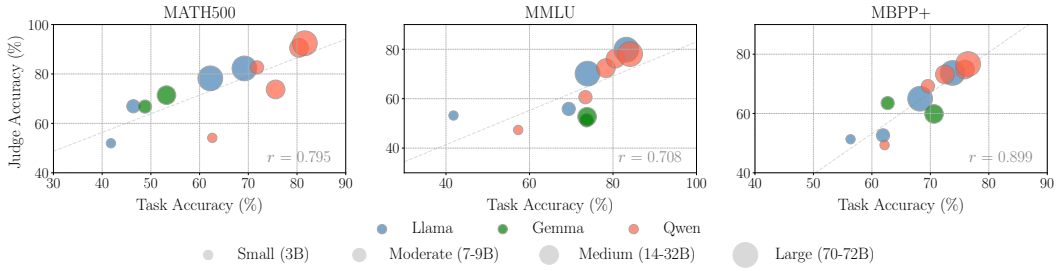

Figure 1: Correlation between judge accuracy (LLMs as evaluators) and task accuracy (LLMs as generators) measured by Pearson correlation coefficient r (Cohen et al., 2009). Each circle represents one LLM, with the size and color denoting model scale and family, respectively. The strong positive correlation between judge accuracy and task accuracy indicates that strong generators are generally accurate evaluators. We provide full results in Tables 7 to 9.

2024); (3) Gemma-2 at 9B and 27B (Team et al., 2024). All models are instruction-tuned versions by default. We also adopt long CoT reasoning models which will be discussed in Section 4.

Evaluatees. We use a fixed set of seven evaluatee models, encompassing both weaker models and strong proprietary ones that represent a broad range of capabilities. This set includes Llama-3.2-1B, Gemma-2-2B, Mistral-7B (Jiang et al., 2023), Mistral-Small, Phi-3.5 (Abdin et al., 2024), and two proprietary models GPT-3.5-Turbo and GPT-4o. All models are also instruction-tuned versions. For all evaluators and evaluatees, we generate verdicts and responses in zero-shot, using greedy decoding by default.

2.3 Tasks

Mathematical reasoning. To assess whether LLM evaluators can identify correct step-by-step solutions for math word problems, we use the MATH500 datasets (Lightman et al., 2023), a curated subset of 500 problems selected from the full MATH dataset introduced by Hendrycks et al. (2021), with accuracy as the evaluation metric.

Factual knowledge. In addition, we employ the popular MMLU (Hendrycks et al., 2020) benchmark, evaluated by accuracy, to test whether LLM evaluators can identify accurate answers for factoid questions—a common and foundational capability in more complex queries involving world knowledge.

Code generation. We adopt the popular code generation benchmark, MBPP+ (Liu et al., 2023), which is an enhanced version of the original MBPP (Austin et al., 2021) with more robust test cases. The results are evaluated using Pass@1.

More implementation details and prompts are provided in Appendices A and C.

3 Strong Models Prefer Themselves Mostly Legitimately

To understand whether LLM evaluators prefer their own generations due to objective quality, we need to quantify the judge model's performance both as a generator and an evaluator. We use the task metrics in Section 2.3 to measure the generator performance, and define the following judge accuracy metric to evaluate the judge performance:

$$\text{Judge\_Acc}_{\mathcal{J}} = \frac{1}{|\mathcal{S}_{\mathcal{G}}|} \sum_{\mathcal{G} \in \mathcal{S}_{\mathcal{G}}} \frac{1}{|\mathcal{D}_{\text{diff}}|} \sum_{x \in \mathcal{D}_{\text{diff}}} \mathbb{1}\left\{\mathcal{J}^*(x, y_{\mathcal{J}}, y_{\mathcal{G}}) = y^*\right\}, \tag{3}$$

where $\mathcal{D}_{\text{diff}}$ refers to a differential subset of $\mathcal{D}$ that includes only instances where either $y_{\mathcal{J}}$ or $y_{\mathcal{G}}$ is correct, but not both, with y^* denoting the correct one. By focusing on $\mathcal{D}_{\text{diff}}$, we ensure the judge's task is unambiguously identifying the correct response, rather than choosing

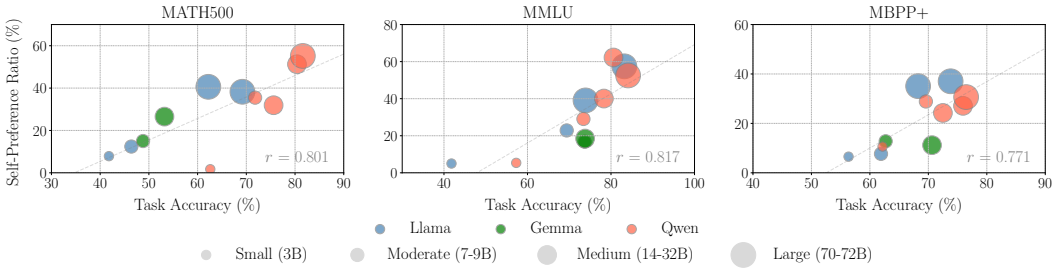

Figure 2: Correlation between self-preference ratio and task accuracy. The clear positive correlation suggests that better generators typically prefer themselves more as evaluators. We provide full results in Tables 1, 3 and 5.

based on stylistic preferences (when both are correct) or facing an ill-defined accuracy task (when both are incorrect).

3.1 Better Generators Are Generally Better Evaluators

Figure 1 demonstrates the correlation between task accuracy and evaluation (judge) accuracy measured with the Pearson correlation coefficient across three benchmark tasks: MATH500, MMLU, and MBPP+. Each point represents a model, with colors distinguishing different model families (Llama, Gemma, and Qwen) and marker sizes indicating model scale. The results reveal a clear positive correlation between task accuracy and judge accuracy, with an r value of 0.795, 0.708, and 0.899 for MATH500, MMLU, and MBPP+, respectively. Our results suggest that models capable of generating more accurate responses are also typically better evaluators.

Scaling and performance. Larger models demonstrate greater reliability as evaluators. For example, Qwen2.5-72B and Llama-3.3-70B exhibit significantly higher evaluation accuracy compared to their smaller counterparts. While smaller models also exhibit a positive correlation, their evaluation accuracy often plateaus at lower levels, implying that scaling improves evaluation capabilities alongside generation abilities.

Implications for tasks. In mathematical reasoning, models that solve problems accurately are more adept at recognizing correct solutions from others. Similarly, in code generation, models that produce correct code are better at identifying bugs and errors in peer code. In factual knowledge tasks, models with higher factual accuracy are more reliable at distinguishing their correct answers from incorrect ones. The consistent strong correlation across all three benchmarks reinforces our findings.

Overall, this relationship partly provides justification for the reliability of LLM evaluators, especially when using the most performant, state-of-the-art models for evaluation.

3.2 Strong Evaluators Favor Themselves Mostly Because They Are Better

Figure 2 illustrates the correlation between model's task accuracy as generators and self-preference ratio (Equation (2)) as evaluators. Similar to Section 3.1, the results also reveal a clear positive correlation between task accuracy and self-preference ratio, with an r value of 0.801, 0.817, and 0.771 for MATH500, MMLU, and MBPP+, respectively. The results indicate that models with superior task performance tend to exhibit stronger self-preference. Similarly, scaling up model size amplifies both task performance and self-preference, and this effect aligns with their superior judge accuracy.

Legitimate self-preference. Collectively, the findings of Figure 1 and 2 suggest that the higher degree of self-preference in stronger models is primarily driven by the objective quality of their outputs (evident by their higher judge accuracy), and less of the potential artifact of bias. Since stronger models generate more accurate responses, their preference for

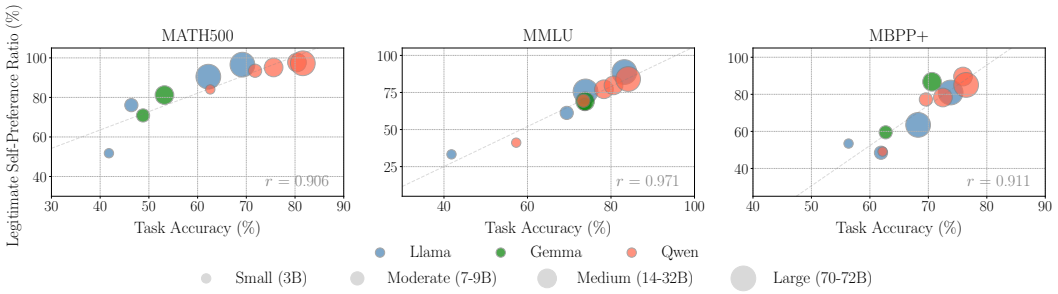

Figure 3: Correlation between legitimate self-preference ratio and task accuracy. The consistent positive correlation indicates that when strong models favor themselves, they are mostly objectively correct. We provide full results in Tables 10 to 12.

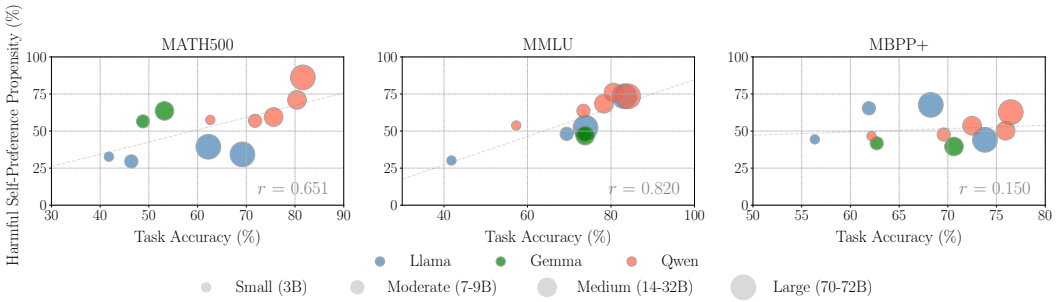

Figure 4: Correlation between harmful self-preference propensity and task accuracy. The positive correlation, in particular on MATH500 and MMLU, implies that when strong models are objectively incorrect, they prefer themselves more often. We provide full results in Tables 2, 4 and 6.

their own outputs could be largely justified, that is, the behavior of legitimate self-preference. We further quantified such behavior by defining *legitimate self-preference ratio* (LSPR) of a judge model $\mathcal{J}$ as follows:

$$\text{LSPR}_{\mathcal{J}} = \frac{1}{|\mathcal{S}_{\mathcal{G}}|} \sum_{\mathcal{G} \in \mathcal{S}_{\mathcal{G}}} \frac{\sum_{x \in \mathcal{D}_{\text{diff}}} \mathbb{1}\left\{\mathcal{J}^*(x, y_{\mathcal{J}}, y_{\mathcal{G}}) = y_{\mathcal{J}} \text{ and } y^* = y_{\mathcal{J}}\right\}}{\sum_{x \in \mathcal{D}_{\text{diff}}} \mathbb{1}\left\{\mathcal{J}^*(x, y_{\mathcal{J}}, y_{\mathcal{G}}) = y_{\mathcal{J}}\right\}}, \tag{4}$$

which quantifies the degree of legitimate self-preference (*i.e.*, $y_{\mathcal{J}}$ is preferred and correct).

As shown in Figure 3, we observe a clear trend demonstrating that an LLM's capability as a task performer positively correlates with its legitimate self-preference ratio. This suggests that as models become more powerful, the proportion of self-preference that is legitimate—where the model favors its own outputs when they are objectively better—also increases. Across different model families (Llama, Gemma, and Qwen) and parameter scales, larger models consistently exhibit higher LSPR values. In particular, Qwen-2.5-70B and Llama-3-70B achieve LSPR of 96.57% and 95.16% on MATH500, respectively. Similarly on MBPP+ and MMLU, Qwen-2.5-70B and Llama-3-70B record an LSPR of 80.98% $\sim$ 88.78%. Overall, the results suggest that strong models favor themselves mostly legitimately.

4 Generating (Long) CoT Reduces Harmful Self-Preference

4.1 Harmful Self-Preference in LLM Evaluators

While our earlier findings suggest self-preference in capable models is often benign, reflecting their genuinely higher output quality, a critical question remains regarding evaluation reliability: What occurs when the evaluator model $\mathcal{J}$ itself generates an objectively incorrect

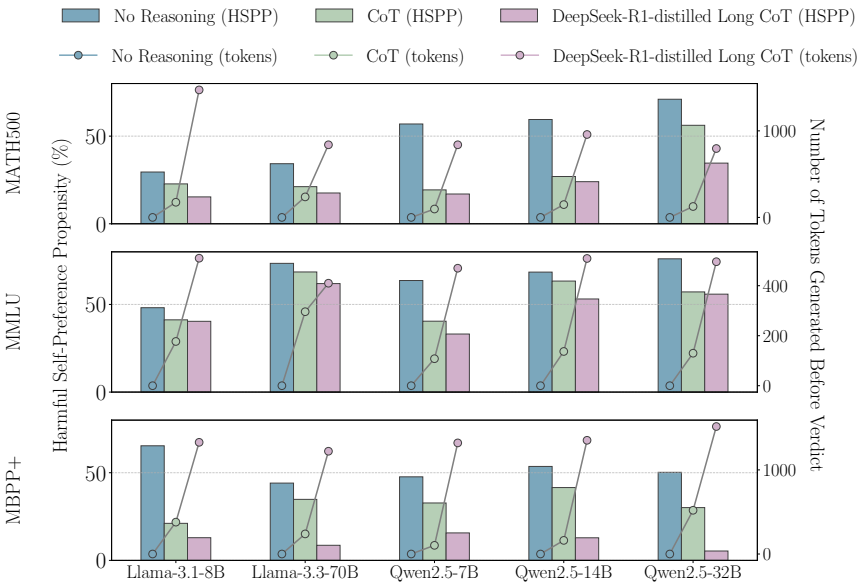

Figure 5: The harmful self-preference propensity at varying levels of evaluator reasoning. When being objectively incorrect, evaluators can achieve more accurate verdict by generating more reasoning tokens prior to their judgments. We provide full results in Tables 13 to 18.

response $y_{\mathcal{J}}$, while an alternative response $y_{\mathcal{G}}$ is correct? To quantify the tendency for bias in these potentially harmful situations, we introduce the *harmful self-preference propensity* (HSPP) for a judge model $\mathcal{J}$ as

$$\text{HSPP}_{\mathcal{J}} = \frac{1}{|\mathcal{S}_{\mathcal{G}}|} \sum_{\mathcal{G} \in \mathcal{S}_{\mathcal{G}}} \frac{\sum_{x \in \mathcal{D}_{\text{diff}}} \mathbb{1}\left\{\mathcal{J}^*(x, y_{\mathcal{J}}, y_{\mathcal{G}}) = y_{\mathcal{J}} \text{ and } y^* = y_{\mathcal{G}}\right\}}{\sum_{x \in \mathcal{D}_{\text{diff}}} \mathbb{1}\left\{y^* = y_{\mathcal{G}}\right\}}, \tag{5}$$

which characterizes the tendency of an LLM evaluator to prefer its own incorrect generation over objectively better ones. A higher HSPP indicates a greater tendency towards harmful bias when the evaluator itself has erred.

Results. As observed in Figure 4, when evaluators' responses are objectively worse, there exists a clear positive correlation between task performance and the harmful self-preference propensity. In other words, stronger models—those with a higher task accuracy—tend to exhibit greater harmful self-preference when evaluating cases where their own outputs are incorrect but the alternative response is correct. This trend is particularly pronounced in larger models. Notably, the most performant model, Qwen2.5-72B, exhibits an HSPP of 86% on MATH500 and 73% on MMLU, significantly higher than its overall SPR (Figure 2) of 55% and 52%, respectively. Although MBPP+ shows a weaker positive correlation, large models like Llama-3.1-70B and Qwen2.5-72B still favor themselves more than smaller ones, with HSPP values ranging from approximately 50% to 75%, significantly higher than their overall SPR (Figure 2), which remains below 40%.

In sum, such model behavior presents a potential safety challenge: as models become more capable, they are also more confident in their own responses, even when they are wrong. This overconfidence could lead to biased evaluation frameworks where stronger models dismiss superior responses from other models, potentially reinforcing flawed or suboptimal outputs for AI oversight.

4.2 Mitigating Harmful Self-Preference Bias via CoT

Compared to recognizing accurate responses, identifying its own mistakes and overriding its initial understanding may require a deeper level of analysis and reasoning for LLM evaluators. With recent advances in reasoning-enhanced, long CoT models, we aim to

investigate the impact of generating reasoning traces for mitigating harmful self-preference. To this end, we experiment with three LLM evaluator settings: no reasoning tokens, standard CoT reasoning, and long CoT reasoning.

No reasoning. The same setting adopted in our previous experiments, where the evaluator is instructed to directly output the corresponding verdict label.

Standard CoT reasoning. The evaluator generates a step-by-step reasoning chain before arriving at its verdict. Both no-reasoning-token setup and standard-CoT-reasoning settings adopt the same underlying instruction-tuned model.

Long CoT reasoning. In this setting, we employ 5 distilled reasoning models with matching backbones, including DeepSeek-R1-Distill-Llama-8B/70B and DeepSeek-R1-Distill-Qwen-7B/14B/32B, from DeepSeek-R1 (Guo et al., 2025). The reasoning model evaluator naturally generates a detailed multi-step reasoning trace before deriving at its verdict.

Results. The results are presented in Figure 5. As shown on both MATH500 and MMLU, generating reasoning traces substantially reduces harmful self-preference across all models. The no-reasoning-token setting exhibits the highest HSPP, and introducing CoT reasoning mitigate the issue to a noticeable degree, lowering harmful self-preference. Reasoning-enhanced models with long CoT further amplify this mitigation, consistently achieving the lowest HSPP across all models.

The trend suggests that reasoning-enhanced evaluations encourage models to more accurately reassess their initial understanding and consider alternative responses more carefully. Interestingly, the relative reduction in HSPP is more pronounced on MATH500 and MBPP+ compared to those on MMLU, possibly indicating that reasoning-intensive tasks benefit more from reasoning-driven evaluations. Overall, these findings underscore the potential of reasoning-enhanced evaluation strategies to improve the reliability of LLM evaluators.

5 Related Work

LLM-based evaluations. Extensive works have explored leveraging LLMs not only as generators, but also as evaluators (Lin & Chen, 2023; Zhu et al., 2023; Ankner et al., 2024; Wei et al., 2025). Early work like LLM-Eval (Lin & Chen, 2023) introduces a unified approach that prompts an LLM to assess multiple quality dimensions of open-domain conversations, reducing costly human annotations. JudgeLM (Zhu et al., 2023) further propose fine-tuning LLMs specifically to act as judges, and demonstrates that with appropriate training, LLMs can serve as scalable judges and achieve a high agreement rate with human judgments. More recently, Liu et al. (2025) investigate improving reward modeling for general queries by scaling inference compute, and propose DeepSeek-GRM. However, it is observed that LLM-based evaluators are often biased (Zheng et al., 2023b; Dubois et al., 2024; Vu et al., 2024; Goel et al., 2025; Kenton et al., 2024). For example, Dubois et al. (2024) employ a regression-based control mechanism to mitigate the length bias prevalent in LLM evaluators. Vu et al. (2024) reduce response ordering bias and length bias in LLM evaluators by training on a diverse set of quality assessment tasks. In this work, we present an in-depth analysis of self-preference bias in LLMs and provide empirical insights for improving the reliability of LLM-based evaluations.

Self-preference bias in LLM evaluators. Several studies have highlighted a prevalent *self-preference bias* (Bai et al., 2023; Li et al., 2024b; Ye et al., 2024), also known as self-enhancement bias (Zheng et al., 2023b)), where LLM evaluators favor their own generations and potentially lead to biased evaluation (Bitton et al., 2023; Liu et al., 2024). Notably, Koo et al. (2024) examine this from a cognitive aspect but do not provide an explanation for the underlying causes. Other works, such as Xu et al. (2024) and Stureborg et al. (2024), propose quantified metrics (*e.g.*, scalar scores) to measure the degree of self-preference bias, while Panickssery et al. (2024) discovered a linear correlation between self-recognition capability and the strength of self-preference bias of LLM evaluators. However, many of them have either constrained their analysis to a limited number of models (Zheng et al.,

2023b; Li et al., 2024b; Xu et al., 2024) or evaluated LLMs under setups where evaluators and evaluatees are drawn from the same model pool (Panickssery et al., 2024; Wataoka et al., 2024), resulting in different evaluators being applied to different sets of evaluatees and obscuring the underlying causes of bias. In contrast, we introduce a controlled setup with designated sets of evaluators and evaluatees, ensuring a consistent assessment across models. This enables us to conduct a comprehensive, large-scale analysis across diverse model families and sizes, yielding more generalizable and systematic insights into the potential drivers and mitigations of the (harmful) self-preference.

6 Discussions

Extending insights beyond verifiable tasks. While our work leverages verifiable tasks to objectively quantify self-preference, the insights provide a crucial foundation for understanding this phenomenon in more subjective, real-world scenarios. First, the underlying mechanisms driving self-preference—ranging from internal model probabilities favoring familiar generation styles (Zheng et al., 2023b; Panickssery et al., 2024; Feuer et al., 2024; Li et al., 2024a), to potential artifacts from alignment training (Leike et al., 2018; Lee et al., 2023)—are inherent properties of the model's architecture and training regime, regardless of whether the evaluation setup is objective or subjective. This suggests that the self-preference patterns in verifiable tasks will likely persist in subjective contexts as well. Second, many complex, real-world tasks often embed objectively verifiable components. For instance, evaluating a persuasive essay involves not only judging argumentation style but also verifying factual claims and logical consistency. Our findings on how self-preference manifests in verifiable tasks offer direct clues of potential reliability issues when LLM evaluators are used for composite, subjective tasks.

Practical recommendations for robust LLM-based evaluation. Our findings offer several practical recommendations for more reliable real-world LLM-based evaluation systems. First, before deploying an LLM as an evaluator for a specific task or domain, its generative performance on the target domain should be rigorously assessed. This pre-assessment helps select models that are more likely to provide accurate judgments and exhibit more legitimate (rather than harmful) preference. Second, our results highlight two complementary pathways to mitigate harmful self-preference bias: (1) Model scaling, which employs larger, more capable evaluators. This primarily boosts reliability by improving generative performance, thus shrinking the set of instances where the model fails and harmful bias can occur. (2) Implementing inference-time scaling (*e.g.*, CoT reasoning) effectively reduce the propensity for harmful bias on those remaining error instances. Finally, for large-scale systems covering diverse domains, consider deploying a roster of specialized LLM evaluators. Evaluation requests could then be dynamically routed to the model with the highest generative capability (and thus, likely highest evaluation accuracy) in the specific domain.

Implications for scalable oversight. A key motivation for LLM-based evaluation is achieving *scalable oversight* (Bai et al., 2022; Bowman et al., 2022; Kenton et al., 2024; Goel et al., 2025)—automating evaluation pipelines for complex domains like competition math or advanced coding, where human evaluation is costly or requires specialized expertise. Before deploying LLMs in such critical roles, their reliability must be thoroughly validated. However, the inherent complexity in these tasks that necessitates AI oversight also makes obtaining human-annotated ground truth for validation exceptionally challenging. Our finding that "better generators are better evaluators" on simpler yet still verifiable benchmarks provides a crucial bridge for establishing trust: We can have greater confidence in evaluations from models known to excel as generators, particularly in their areas of strength, potentially reducing the need for exhaustive human review. Complementing this, our finding that harmful self-preference bias persists, particularly when models fail on specific task instances, enables the strategic targeting of human intervention. We may focus review efforts on evaluations from models known to struggle with certain task types, or on particularly challenging instances where self-preference bias is most likely to emerge. This targeted approach enables more efficient allocation of limited human review resources compared to uniform human oversight, making the entire evaluation system more scalable while maintaining reliability.

7 Conclusion & Limitation

In this work, we investigate self-preference in LLMs using verifiable benchmarks with objective ground-truth reference. Our experiments suggest that self-preference often reflects genuine output quality in more capable models. Stronger models tend to be more accurate judges, with their preferences frequently aligned with objective correctness. However, harmful self-preference persists when models favor an objectively incorrect response by themselves. To address this, we explore inference-time scaling strategies and demonstrate that, while being objectively worse as generators, evaluators can achieve more accurate verdict by generating more reasoning tokens. Overall, our study provides deeper insights into understanding and improving LLM-based evaluation, although several directions remain open for future exploration, and we discuss them as follows.

Biases and hacking other than self-preference. While we focus on self-preference bias, LLM evaluators may also exhibit other forms of bias, such as length bias (Saito et al., 2023; Zheng et al., 2023b), order bias (Zheng et al., 2023a; Wang et al., 2023; Pezeshkpour & Hruschka, 2024; Wei et al., 2024; Shi et al., 2024), or stylistic bias (Chen et al., 2024a; Ye et al., 2024), which undermine the fairness of LLM-based evaluations. In addition, Zheng et al. (2025) uncover potential hacking in LLM-as-a-Judge benchmarking, showing that simple "null" models that always produce constant outputs can cheat automatic evaluation and achieve top-performing win rates. Collectively, these findings highlight the need for more comprehensive audits to ensure robustness and reliability for LLM-based evaluations.

Diverse inference-time scaling strategies. Our experiments on varying levels of CoT reasoning length represent a subset of inference-time scaling methods (Welleck et al., 2024). Future work can explore a broader space of techniques for scaling evaluation-time compute, such as self-consistency decoding (Wang et al., 2022), Best-of-N via repeated sampling (Brown et al., 2024; Chen et al., 2024b), and multi-agent verification (Lifshitz et al., 2025). A notable concurrent work by Kim et al. (2025) explores using reasoning models as process evaluators and demonstrates its superiority over process reward models. In comparison, we focus on investigating self-evaluation in setups analogous to LLM-as-a-Judge.

Broader LLM-based evaluation formats. While our experiments investigate self-preference strictly as a model favoring its exact own output, a more relaxed definition may include preference toward outputs from models within the same family. Exploring this broader notion could uncover subtler forms of bias that arise from shared training signals or architectural similarities. To enable consistent measurement of self-preference behaviors, we adopt a controlled setup that separates evaluators and evaluatees and focuses on discrete, pairwise evaluation. However, applications such as scalar scoring or ranking might involve pointwise evaluation (Lee et al., 2023; Vu et al., 2024; Zhang et al., 2024). Extending our methodology to these alternative formats would help generalize our findings and offer a more comprehensive understanding of self-preference in LLM-based evaluations.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Zachary Ankner, Mansheej Paul, Brandon Cui, Jonathan D Chang, and Prithviraj Ammanabrolu. Critique-out-loud reward models. *arXiv preprint arXiv:2408.11791*, 2024.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Yushi Bai, Jiahao Ying, Yixin Cao, Xin Lv, Yuze He, Xiaozhi Wang, Jifan Yu, Kaisheng Zeng, Yijia Xiao, Haozhe Lyu, et al. Benchmarking foundation models with language-model-as-an-examiner. *Advances in Neural Information Processing Systems*, 36:78142–78167, 2023.

Yonatan Bitton, Hritik Bansal, Jack Hessel, Rulin Shao, Wanrong Zhu, Anas Awadalla, Josh Gardner, Rohan Taori, and Ludwig Schimdt. Visit-bench: a benchmark for vision-language instruction following inspired by real-world use. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pp. 26898–26922, 2023.

Samuel R Bowman, Jeeyoon Hyun, Ethan Perez, Edwin Chen, Craig Pettit, Scott Heiner, Kamilė Lukošiūtė, Amanda Askell, Andy Jones, Anna Chen, et al. Measuring progress on scalable oversight for large language models. *arXiv preprint arXiv:2211.03540*, 2022.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv*, 2024. doi: 10.48550/arxiv.2407.21787.

Guiming Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. Humans or LLMs as the judge? a study on judgement bias. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8301–8327, 2024a.

Lingjiao Chen, Jared Quincy Davis, Boris Hanin, Peter Bailis, Ion Stoica, Matei Zaharia, and James Zou. Are more LM calls all you need? towards the scaling properties of compound AI systems. *arXiv*, 2024b. doi: 10.48550/arxiv.2403.02419.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Israel Cohen, Yiteng Huang, Jingdong Chen, Jacob Benesty, Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson correlation coefficient. *Noise reduction in speech processing*, pp. 1–4, 2009.

Yann Dubois, Percy Liang, and Tatsunori Hashimoto. Length-controlled alpacaeval: A simple debiasing of automatic evaluators. In *First Conference on Language Modeling*, 2024.

Benjamin Feuer, Micah Goldblum, Teresa Datta, Sanjana Nambiar, Raz Besaleli, Samuel Dooley, Max Cembalest, and John P Dickerson. Style outweighs substance: Failure modes of LLM judges in alignment benchmarking. *arXiv preprint arXiv:2409.15268*, 2024.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*, 2023.

Shashwat Goel, Joschka Struber, Ilze Amanda Auzina, Karuna K. Chandra, Ponnurangam Kumaraguru, Douwe Kiela, Ameya Prabhu, Matthias Bethge, and Jonas Geiping. Great models think alike and this undermines AI oversight. *CoRR*, abs/2502.04313, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv*, 2020. doi: 10.48550/arxiv.2009.03300.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L'elio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7B. *ArXiv*, abs/2310.06825, 2023.

Zachary Kenton, Noah Y. Siegel, János Kramár, Jonah Brown-Cohen, Samuel Albanie, Jannis Bulian, Rishabh Agarwal, David Lindner, Yunhao Tang, Noah D. Goodman, and Rohin Shah. On scalable oversight with weak LLMs judging strong LLMs. In *NeurIPS*, 2024.

Seungone Kim, Ian Wu, Jinu Lee, Xiang Yue, Seongyun Lee, Mingyeong Moon, Kiril Gashteovski, Carolin Lawrence, Julia Hockenmaier, Graham Neubig, et al. Scaling evaluation-time compute with reasoning models as process evaluators. *arXiv preprint arXiv:2503.19877*, 2025.

Ryan Koo, Minhwa Lee, Vipul Raheja, Jong Inn Park, Zae Myung Kim, and Dongyeop Kang. Benchmarking cognitive biases in large language models as evaluators. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 517–545, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, et al. RLAIF vs. RLHF: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023.

Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022.

Tianle Li, Anastasios Angelopoulos, and Wei-Lin Chiang. Does style matter? disentangling style and substance in chatbot arena, August 2024a. URL https://blog.lmarena.ai/blog/2024/style-control/.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*, 2024b.

Shalev Lifshitz, Sheila A McIlraith, and Yilun Du. Multi-agent verification: Scaling test-time compute with goal verifiers. In *Workshop on Reasoning and Planning for Large Language Models*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Yen-Ting Lin and Yun-Nung Chen. LLM-Eval: Unified multi-dimensional automatic evaluation for open-domain conversations with large language models. *CoRR*, abs/2305.13711, 2023.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems*, 36:21558–21572, 2023.

Yiqi Liu, Nafise Sadat Moosavi, and Chenghua Lin. LLMs as narcissistic evaluators: When ego inflates evaluation scores. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 12688–12701, 2024.

Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. Inference-time scaling for generalist reward modeling, 2025. URL https://arxiv.org/abs/2504.02495.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36: 46534–46594, 2023.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. LLM evaluators recognize and favor their own generations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=4NJBV6Wp0h.

Pouya Pezeshkpour and Estevam Hruschka. Large language models sensitivity to the order of options in multiple-choice questions. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 2006–2017, 2024.

Keita Saito, Akifumi Wachi, Koki Wataoka, and Youhei Akimoto. Verbosity bias in preference labeling by large language models. *arXiv preprint arXiv:2310.10076*, 2023.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*, 2022.

Sumuk Shashidhar, Abhinav Chinta, Vaibhav Sahai, Zhenhailong Wang, and Heng Ji. Democratizing LLMs: An exploration of cost-performance trade-offs in self-refined open-source models. *arXiv preprint arXiv:2310.07611*, 2023.

Lin Shi, Chiyu Ma, Wenhua Liang, Weicheng Ma, and Soroush Vosoughi. Judging the judges: A systematic investigation of position bias in pairwise comparative assessments by LLMs. *arXiv preprint arXiv:2406.07791*, 2024.

Kumar Shridhar, Koustuv Sinha, Andrew Cohen, Tianlu Wang, Ping Yu, Ram Pasunuru, Mrinmaya Sachan, Jason Weston, and Asli Celikyilmaz. The art of LLM refinement: Ask, refine, and trust. *arXiv preprint arXiv:2311.07961*, 2023.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021, 2020.

Rickard Stureborg, Dimitris Alikaniotis, and Yoshi Suhara. Large language models are inconsistent and biased evaluators. *arXiv preprint arXiv:2405.01724*, 2024.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Tu Vu, Kalpesh Krishna, Salaheddin Alzubi, Chris Tar, Manaal Faruqui, and Yun-Hsuan Sung. Foundational autoraters: Taming large language models for better automatic evaluation. In *EMNLP*, pp. 17086–17105. Association for Computational Linguistics, 2024.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. *arXiv*, 2023. doi: 10.48550/arxiv.2305.17926.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv*, 2022. doi: 10.48550/arxiv.2203.11171.

Koki Wataoka, Tsubasa Takahashi, and Ryokan Ri. Self-preference bias in LLM-as-a-judge. In *Neurips Safe Generative AI Workshop*, 2024.

Sheng-Lun Wei, Cheng-Kuang Wu, Hen-Hsen Huang, and Hsin-Hsi Chen. Unveiling selection biases: Exploring order and token sensitivity in large language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 5598–5621, 2024.

Zhepei Wei, Wei-Lin Chen, and Yu Meng. InstructRAG: Instructing retrieval-augmented generation via self-synthesized rationales. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=P1qhkp8gQT.

Sean Welleck, Amanda Bertsch, Matthew Finlayson, Hailey Schoelkopf, Alex Xie, Graham Neubig, Ilia Kulikov, and Zaid Harchaoui. From decoding to meta-generation: Inference-time algorithms for large language models. *arXiv*, 2024. doi: 10.48550/arxiv.2406.16838.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Jeff Wu, Long Ouyang, Daniel M Ziegler, Nisan Stiennon, Ryan Lowe, Jan Leike, and Paul Christiano. Recursively summarizing books with human feedback. *arXiv preprint arXiv:2109.10862*, 2021.

Wenda Xu, Guanglei Zhu, Xuandong Zhao, Liangming Pan, Lei Li, and William Wang. Pride and prejudice: LLM amplifies self-bias in self-refinement. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15474–15492, 2024.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, et al. Justice or prejudice? quantifying biases in LLM-as-a-judge. In *Neurips Safe Generative AI Workshop*, 2024.

Zhiqiang Yuan, Junwei Liu, Qiancheng Zi, Mingwei Liu, Xin Peng, and Yiling Lou. Evaluating instruction-tuned large language models on code comprehension and generation. *arXiv preprint arXiv:2308.01240*, 2023.

Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. Evaluating large language models at evaluating instruction following. In *The Twelfth International Conference on Learning Representations*, 2024.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*, 2024.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. Large language models are not robust multiple choice selectors. *arXiv preprint arXiv:2309.03882*, 2023a.

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, 2023b.

Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Jing Jiang, and Min Lin. Cheating automatic LLM benchmarks: Null models achieve high win rates. In *The Thirteenth International Conference on Learning Representations*, 2025.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. JudgeLM: Fine-tuned large language models are scalable judges. *CoRR*, abs/2310.17631, 2023.

A Implementation Details

Models. We adopt the following model checkpoints/versions from Huggingface (Wolf et al., 2019) and OpenAI API[3] for our experiments:

- Llama family: meta-llama/Llama-3.2-1B-Instruct, meta-llama/Llama-3.2-3B-Instruct, meta-llama/Llama-3.1-8B-Instruct, meta-llama/Llama-3.1-70B-Instruct, meta-llama/Llama-3.3-70B-Instruct
- Gemma family: google/gemma-2-2b-it, google/gemma-2-9b-it, google/gemma-2-27b-it
- Qwen family: Qwen/Qwen2.5-3B-Instruct, Qwen/Qwen2.5-7B-Instruct, Qwen/Qwen2.5-14B-Instruct, Qwen/Qwen2.5-32B-Instruct, Qwen/Qwen2.5-72B-Instruct
- DeepSeek-R1-Distill family: deepseek-ai/DeepSeek-R1-Distill-Llama-8B, deepseek-ai/DeepSeek-R1-Distill-Llama-70B, deepseek-ai/DeepSeek-R1-Distill-Qwen-7B, deepseek-ai/DeepSeek-R1-Distill-Qwen-14B, deepseek-ai/DeepSeek-R1-Distill-Qwen-32B
- Mistral family: mistralai/Mistral-7B-Instruct-v0.3, mistralai/Mistral-Small-Instruct-2409
- Phi family: microsoft/Phi-3.5-mini-instruct
- GPT family: gpt-3.5-turbo-0125, gpt-4o-2024-11-20

Response generation. For generating responses, we adopt `temperature` = 0 (*i.e.*, greedy decoding) for all instruction-tuned models, including OpenAI API models; we adopt `temperature` = 0.6 for all DeepSeek-R1-distilled reasoning models as recommended in the model document to prevent endless repetitions or incoherent outputs. Also, to prevent potential length biases in evaluator, for reasoning models we preserve only the responses after the "<\think>" token as the answer to be evaluated (*i.e.*, the long, verbose reasoning traces within "<think>" and "<\think>" are not presented to the evaluators). Note that the responses after the "<\think>" token still contain concise explanations and rationales similar to the response from instruction-tuned, non-reasoning models.

Verdict generation. For generating verdicts, we also adopt `temperature` = 0 for all non-reasoning models and `temperature` = 0.6 for reasoning models. By default, our experiments employ `max_tokens` = 1, and we obtain the verdict by instructing the model to directly output a label: "A", "T", or "B", corresponding to which assistant's answer is better or they are relatively the same in quality. Specifically, we examine the label set for their corresponding candidate tokens and select the token among the three with the highest assigned logit as the evaluator's verdict.

For experiments in Section 4, the no reasoning token setting adopts the above-described configuration. For the standard CoT reasoning setting, the model is allowed to freely generate its judgment in natural language with a reasoning chain, before ending its response with a verdict (*e.g.*, "My final verdict is {verdict}"). The verdict is then parsed into the corresponding label. For long CoT reasoning, the model is instructed in the same way as the standard CoT reasoning setting, allowing it to freely generate the judgment.

Other details. We perform all model inference (except proprietary OpenAI API models) using the vLLM library (Kwon et al., 2023). The evaluation script for task accuracy is adopted from lm-evaluation-harness and Lewkowycz et al. (2022) for MATH500; code_eval and Chen et al. (2021) for MBPP+. We calculate the token length in Figure 5 using GPT-2 tokenizer from tiktoken. The majority of our experiments are conducted on cloud computing infrastructure with access to 8 NVIDIA A100 GPUs per instance.

We randomly sample 1K instances from the full MMLU test set for our MMLU experiments to ensure computational efficiency, given our large set of evaluators and evaluatees. Our

[3] https://platform.openai.com/docs/models

preliminary studies indicate that such a 1K subset is sufficient to produce stable results, and further expansion has negligible impact on the outcomes.

B Full Experimental Results

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	4.2	5.8	1.0	12.0	11.0	17.2	3.8	7.9
Llama-3.1-8B	20.6	12.8	7.2	5.2	22.8	11.6	6.6	12.4
Llama-3.1-70B	68.4	36.4	34.6	12.6	59.0	56.4	16.8	40.6
Llama-3.3-70B	66.4	34.6	31.8	9.2	56.8	51.0	18.0	38.3
Gemma-2-9B	25.2	9.0	11.4	6.2	28.2	19.8	5.2	15.0
Gemma-2-27B	39.6	20.8	20.4	8.2	45.4	38.6	12.8	26.5
Qwen2.5-3B	1.6	1.2	2.4	1.2	2.0	1.2	2.2	1.7
Qwen2.5-7B	61.4	32.4	26.4	6.8	54.2	53.8	13.4	35.5
Qwen2.5-14B	53.4	27.6	28.6	8.4	55.6	33.6	16.2	31.9
Qwen2.5-32B	82.6	43.8	51.4	11.8	70.8	71.2	27.2	51.3
Qwen2.5-72B	86.4	50.6	56.4	15.8	72.6	73.2	31.2	55.2

Table 1: Full SPR results of MATH500. The numbers in Figure 2 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	38.1	34.4	25.4	29.2	20.0	52.2	29.6	32.7
Llama-3.1-8B	43.5	32.1	22.8	19.0	43.5	29.2	16.7	29.5
Llama-3.1-70B	50.0	25.0	34.8	20.9	44.4	75.0	25.6	39.4
Llama-3.3-70B	75.0	20.0	26.7	20.0	28.6	41.7	28.0	34.3
Gemma-2-9B	69.2	53.5	53.7	43.9	42.9	90.0	42.7	56.5
Gemma-2-27B	81.8	58.3	50.0	42.1	77.8	81.0	54.1	63.6
Qwen2.5-3B	71.4	57.9	55.0	36.3	71.4	72.7	37.8	57.5
Qwen2.5-7B	66.7	72.7	27.3	51.3	80.0	62.5	38.1	56.9
Qwen2.5-14B	80.0	83.3	71.4	23.3	100.0	16.7	41.7	59.5
Qwen2.5-32B	100.0	100.0	80.0	33.3	75.0	71.4	37.5	71.0
Qwen2.5-72B	100.0	100.0	100.0	53.3	100.0	100.0	50.0	86.2

Table 2: Full HSPP results of MATH500. The numbers in Figure 4 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	2.2	4.3	4.2	8.3	4.5	8.0	3.6	5.0
Llama-3.1-8B	37.2	11.9	19.7	2.6	18.2	52.8	17.6	22.9
Llama-3.1-70B	63.5	22.4	37.5	2.8	39.5	80.6	27.1	39.1
Llama-3.3-70B	74.7	50.8	64.8	12.9	57.2	81.5	60.3	57.5
Gemma-2-9B	26.1	8.7	13.9	2.0	12.2	44.1	10.4	16.8
Gemma-2-27B	24.4	8.8	16.1	5.9	13.5	49.0	11.7	18.5
Qwen2.5-3B	6.9	5.6	6.0	2.8	4.7	6.0	5.5	5.4
Qwen2.5-7B	46.9	11.6	15.8	2.6	27.9	76.9	21.7	29.1
Qwen2.5-14B	63.6	24.2	39.0	5.1	38.1	75.9	34.6	40.1
Qwen2.5-32B	84.9	57.8	66.9	9.2	61.6	88.4	66.7	62.2
Qwen2.5-72B	79.8	35.4	56.1	4.3	48.2	92.9	51.1	52.5

Table 3: Full SPR results of MMLU. The numbers in Figure 2 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	21.5	19.5	26.4	37.8	24.6	59.9	21.3	30.1
Llama-3.1-8B	66.3	39.3	54.3	12.0	34.5	88.0	42.6	48.1
Llama-3.1-70B	77.6	41.7	48.8	10.9	52.5	90.0	44.2	52.2
Llama-3.3-70B	89.7	88.9	77.8	25.0	72.4	94.3	66.0	73.4
Gemma-2-9B	56.9	36.4	42.9	26.1	44.8	88.3	38.1	47.6
Gemma-2-27B	54.9	39.4	51.8	23.7	32.1	87.9	38.5	46.9
Qwen2.5-3B	60.3	47.3	56.7	58.7	62.2	43.0	48.0	53.7
Qwen2.5-7B	81.4	55.2	58.4	13.5	75.0	98.3	63.7	63.6
Qwen2.5-14B	68.4	70.0	80.0	31.3	66.7	98.1	64.6	68.4
Qwen2.5-32B	89.5	77.8	94.0	27.2	73.3	98.0	72.4	76.0
Qwen2.5-72B	91.7	88.9	88.2	20.7	54.2	100.0	70.6	73.5

Table 4: Full HSPP results of MMLU. The numbers in Figure 4 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	6.3	8.2	3.7	6.3	5.8	7.9	7.4	6.5
Llama-3.1-8B	10.3	7.9	6.1	7.7	6.1	6.1	9.5	7.7
Llama-3.1-70B	48.4	31.0	27.5	19.0	52.4	55.6	25.4	37.0
Llama-3.3-70B	52.4	26.2	20.6	15.1	50.3	56.9	24.1	35.1
Gemma-2-9B	16.4	13.2	12.2	8.2	14.3	16.4	8.7	12.8
Gemma-2-27B	18.0	10.3	8.5	4.8	14.0	17.5	5.3	11.2
Qwen2.5-3B	10.8	6.1	12.7	9.3	14.8	10.6	10.1	10.6
Qwen2.5-7B	40.7	26.7	21.2	13.8	34.7	42.1	23.0	28.9
Qwen2.5-14B	40.5	18.0	10.3	2.1	38.6	45.5	14.6	24.2
Qwen2.5-32B	43.9	20.1	15.1	6.3	41.3	46.6	16.4	27.1
Qwen2.5-72B	49.2	24.1	12.7	6.9	48.7	52.9	20.1	30.7

Table 5: Full SPR results of MBPP+. The numbers in Figure 2 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	46.7	55.8	35.6	48.2	29.4	52.4	42.9	44.4
Llama-3.1-8B	42.9	85.3	74.0	73.1	45.5	50.0	86.8	65.4
Llama-3.1-70B	66.7	78.6	71.4	53.5	72.7	66.7	64.0	67.6
Llama-3.3-70B	37.5	53.3	45.5	41.4	64.3	16.7	50.0	44.1
Gemma-2-9B	41.7	45.7	49.1	22.4	46.2	37.5	50.0	41.8
Gemma-2-27B	50.0	23.1	50.0	30.6	41.7	40.0	41.7	39.6
Qwen2.5-3B	50.0	34.5	46.0	58.1	39.3	46.7	51.5	46.6
Qwen2.5-7B	66.7	50.0	44.1	26.8	68.8	50.0	27.3	47.7
Qwen2.5-14B	55.6	56.3	46.4	46.4	61.1	77.8	31.6	53.6
Qwen2.5-32B	50.0	56.3	44.4	28.0	56.3	66.7	50.0	50.2
Qwen2.5-72B	90.0	53.8	50.0	57.9	87.5	50.0	50.0	62.7

Table 6: Full HSPP results of MBPP+. The numbers in Figure 4 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	30.5	49.1	44.1	67.3	54.4	66.0	52.4	52.0
Llama-3.1-8B	70.2	62.8	54.7	74.4	68.9	68.1	69.4	66.9
Llama-3.1-70B	89.6	80.9	75.3	74.4	83.8	84.2	59.5	78.2
Llama-3.3-70B	92.0	85.3	79.9	75.6	87.1	91.4	64.4	82.2
Gemma-2-9B	80.9	68.2	64.7	56.4	66.5	70.6	60.3	66.8
Gemma-2-27B	83.1	68.7	65.2	59.2	78.6	83.5	61.8	71.4
Qwen2.5-3B	48.8	41.4	50.7	64.5	71.1	50.2	52.3	54.1
Qwen2.5-7B	94.6	89.2	70.7	60.6	95.0	96.2	72.4	82.7
Qwen2.5-14B	81.7	69.5	69.2	69.4	90.8	63.0	72.6	73.8
Qwen2.5-32B	97.8	94.0	92.2	75.0	95.7	96.4	82.9	90.6
Qwen2.5-72B	98.8	95.8	92.4	76.4	96.9	98.0	88.7	92.4

Table 7: Full judge accuracy results of MATH500. The numbers in Figure 1 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	58.1	37.7	53.3	60.0	48.2	43.6	71.5	53.2
Llama-3.1-8B	57.7	39.6	42.5	75.6	48.0	74.3	53.3	55.9
Llama-3.1-70B	73.3	53.7	63.0	78.9	72.0	84.4	65.6	70.1
Llama-3.3-70B	87.1	79.9	78.2	68.6	85.0	90.3	69.7	79.8
Gemma-2-9B	51.2	30.8	44.6	67.2	41.3	68.6	54.7	51.2
Gemma-2-27B	56.7	27.8	39.3	69.9	42.5	75.4	57.4	52.7
Qwen2.5-3B	45.9	37.9	45.7	41.9	59.1	52.4	48.4	47.3
Qwen2.5-7B	71.9	31.1	48.3	74.8	64.3	82.5	51.3	60.6
Qwen2.5-14B	83.0	70.8	67.7	64.6	73.7	85.0	61.8	72.4
Qwen2.5-32B	87.9	74.1	72.3	64.5	80.8	89.3	64.6	76.2
Qwen2.5-72B	90.8	72.8	70.6	60.4	85.4	90.8	75.7	78.1

Table 8: Full judge accuracy results of MMLU. The numbers in Figure 1 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	37.3	51.9	60.2	48.9	60.0	46.8	54.1	51.3
Llama-3.1-8B	77.6	53.0	36.8	32.9	60.9	58.8	48.6	52.6
Llama-3.1-70B	80.0	62.8	45.2	47.3	75.0	85.8	58.8	65.0
Llama-3.3-70B	80.9	75.3	63.2	52.1	84.3	94.6	65.3	73.7
Gemma-2-9B	72.5	56.3	53.2	74.4	61.0	71.3	55.7	63.5
Gemma-2-27B	68.7	64.9	40.4	56.0	67.6	72.4	49.0	59.9
Qwen2.5-3B	47.5	54.4	55.1	44.3	49.5	47.1	47.5	49.3
Qwen2.5-7B	82.8	62.0	53.8	64.3	70.6	86.9	64.2	69.2
Qwen2.5-14B	88.0	74.3	60.9	53.7	77.4	88.5	69.4	73.2
Qwen2.5-32B	84.2	74.7	68.0	56.3	85.7	93.1	62.0	74.8
Qwen2.5-72B	91.3	82.3	69.1	42.1	86.7	93.9	71.4	76.7

Table 9: Full judge accuracy results of MBPP+. The numbers in Figure 1 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	85.7	0.0	66.7	16.7	100.0	93.3	0.0	51.8
Llama-3.1-8B	94.0	90.0	83.3	0.0	90.4	95.0	80.0	76.1
Llama-3.1-70B	98.4	93.8	92.0	68.2	98.1	96.7	86.2	90.5
Llama-3.3-70B	98.5	97.5	97.5	93.3	99.4	98.2	91.4	96.6
Gemma-2-9B	98.5	62.5	79.2	16.7	98.3	91.1	50.0	70.9
Gemma-2-27B	97.2	84.1	91.7	37.5	95.6	89.0	73.3	81.2
Qwen2.5-3B	100.0	80.0	100.0	33.3	100.0	100.0	75.0	84.0
Qwen2.5-7B	99.5	95.9	96.5	66.7	99.4	98.9	97.6	93.5
Qwen2.5-14B	99.4	96.2	97.6	75.0	100.0	100.0	97.9	95.2
Qwen2.5-32B	99.0	99.3	98.1	92.3	99.2	98.8	97.1	97.7
Qwen2.5-72B	99.1	100.0	97.8	87.2	99.3	98.6	99.0	97.3

Table 10: Full LSPR results of MATH500. The numbers in Figure 3 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	16.7	66.7	31.3	0.0	45.5	65.7	7.1	33.3
Llama-3.1-8B	71.8	84.6	53.0	16.7	81.5	83.4	37.5	61.2
Llama-3.1-70B	81.1	88.4	81.7	36.4	92.5	87.4	62.7	75.7
Llama-3.3-70B	92.3	97.2	91.3	68.8	96.9	93.7	81.3	88.8
Gemma-2-9B	83.3	88.1	75.0	18.2	90.6	84.7	30.8	67.2
Gemma-2-27B	84.4	84.2	68.5	9.1	95.9	87.7	53.6	69.1
Qwen2.5-3B	51.6	58.3	26.7	13.6	45.0	77.4	15.4	41.2
Qwen2.5-7B	84.4	75.9	68.7	33.3	86.1	86.9	50.6	69.4
Qwen2.5-14B	91.5	91.9	80.7	30.0	94.9	89.1	62.0	77.2
Qwen2.5-32B	90.9	95.5	82.9	35.3	93.8	90.6	69.6	79.8
Qwen2.5-72B	93.3	93.0	87.8	45.0	96.0	92.7	80.6	84.1

Table 11: Full LSPR results of MMLU. The numbers in Figure 3 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.2-3B	85.7	83.3	0.0	0.0	100.0	80.0	25.0	53.4
Llama-3.1-8B	100.0	12.5	50.0	0.0	85.7	80.0	11.1	48.5
Llama-3.1-70B	96.2	56.1	44.0	10.0	85.7	91.9	60.7	63.5
Llama-3.3-70B	97.5	86.7	73.7	30.8	91.8	100.0	86.5	81.0
Gemma-2-9B	96.6	37.5	29.4	14.3	93.3	95.5	50.0	59.5
Gemma-2-27B	100.0	100.0	44.4	66.7	96.3	100.0	100.0	86.8
Qwen2.5-3B	100.0	33.3	14.3	12.5	76.9	66.7	40.0	49.1
Qwen2.5-7B	96.7	82.6	63.2	25.0	91.1	96.3	85.7	77.2
Qwen2.5-14B	97.2	80.0	58.8	40.0	91.8	97.0	83.3	78.3
Qwen2.5-32B	97.2	90.0	78.3	75.0	93.2	97.3	96.3	89.6
Qwen2.5-72B	96.0	90.9	83.3	40.0	92.8	97.6	94.9	85.1

Table 12: Full LSPR results of MBPP+. The numbers in Figure 3 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.1-8B No reasoning	43.5	32.1	22.8	19.0	43.5	29.2	16.7	29.5
Llama-3.1-8B CoT	39.1	19.6	26.3	6.5	30.4	25.0	12.2	22.8
DeepSeek-R1-Distill-Llama-8B	16.7	16.1	14.8	8.3	11.1	27.3	13.0	15.3
Llama-3.3-70B No reasoning	75.0	20.0	26.7	20.0	28.6	41.7	28.0	34.3
Llama-3.3-70B CoT	62.5	20.0	20.0	12.7	0.0	25.0	8.0	21.2
DeepSeek-R1-Distill-Llama-70B	50.0	0.0	44.4	0.0	0.0	0.0	28.6	17.6
Qwen2.5-7B No reasoning	66.7	72.7	27.3	51.3	80.0	62.5	38.1	56.9
Qwen2.5-7B CoT	33.3	9.1	45.5	10.3	20.0	12.5	4.8	19.3
DeepSeek-R1-Distill-Qwen-7B	66.7	7.1	7.1	9.4	0.0	14.3	14.3	17.0
Qwen2.5-14B No reasoning	80.0	83.3	71.4	23.3	100.0	16.7	41.7	59.5
Qwen2.5-14B CoT	60.0	16.7	57.1	13.3	0.0	33.3	8.3	27.0
DeepSeek-R1-Distill-Qwen-14B	60.0	7.7	25.0	11.5	16.7	25.0	22.2	24.0
Qwen2.5-32B No reasoning	100.0	100.0	80.0	33.3	75.0	71.4	37.5	71.0
Qwen2.5-32B CoT	100.0	50.0	100.0	11.1	50.0	57.1	25.0	56.2
DeepSeek-R1-Distill-Qwen-32B	100.0	40.0	50.0	0.0	0.0	25.0	27.3	34.6

Table 13: Full HSPP results of MATH500 at varying levels of evaluator reasoning. The numbers in Figure 5 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7b	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.1-8B No reasoning	66.3	39.3	54.3	12.0	34.5	88.0	42.6	48.1
Llama-3.1-8B CoT	60.5	26.8	50.5	2.8	24.1	86.7	37.2	41.2
DeepSeek-R1-Distill-Llama-8B	46.2	30.0	47.2	10.9	33.8	82.4	30.7	40.2
Llama-3.3-70B No reasoning	89.7	88.9	77.8	25.0	72.4	94.3	66.0	73.4
Llama-3.3-70B CoT	93.1	72.2	75.0	17.1	65.5	97.1	59.6	68.5
DeepSeek-R1-Distill-Llama-70B	70.8	52.6	71.0	24.5	70.0	86.2	58.1	61.9
Qwen2.5-7B No reasoning	81.4	55.2	58.4	13.5	75.0	98.3	63.7	63.6
Qwen2.5-7B CoT	54.2	24.1	29.9	4.1	39.6	96.7	34.5	40.4
DeepSeek-R1-Distill-Qwen-7B	39.8	31.4	32.9	9.4	25.3	63.5	29.5	33.1
Qwen2.5-14B No reasoning	68.4	70.0	80.0	31.3	66.7	98.1	64.6	68.4
Qwen2.5-14B CoT	76.3	60.0	78.0	12.2	63.3	98.1	55.4	63.3
DeepSeek-R1-Distill-Qwen-14B	72.7	37.9	59.3	8.0	63.3	85.1	45.3	53.1
Qwen2.5-32B No reasoning	89.5	77.8	94.0	27.2	73.3	98.0	72.4	76.0
Qwen2.5-32B CoT	78.9	50.0	64.0	12.0	53.3	90.0	51.7	57.1
DeepSeek-R1-Distill-Qwen-32B	63.0	50.0	73.5	18.5	52.2	81.3	52.6	55.9

Table 14: Full HSPP results of MMLU at varying levels of evaluator reasoning. The numbers in Figure 5 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7B	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.1-8B No reasoning	42.9	85.3	74.0	73.1	45.5	50.0	86.8	65.4
Llama-3.1-8B CoT	14.3	23.5	20.0	17.9	31.8	16.7	23.7	21.1
DeepSeek-R1-Distill-Llama-8B	8.3	8.9	15.5	11.6	9.7	22.2	14.3	12.9
Llama-3.3-70B No reasoning	37.5	53.3	45.5	41.4	64.3	16.7	50.0	44.1
Llama-3.3-70B CoT	37.5	40.0	22.7	27.6	71.4	16.7	27.8	34.8
DeepSeek-R1-Distill-Llama-70B	25.0	4.4	7.0	3.2	6.7	7.4	6.5	8.6
Qwen2.5-7B No reasoning	66.7	50.0	44.1	26.8	68.8	50.0	27.3	47.7
Qwen2.5-7B CoT	33.3	40.9	29.4	22.0	50.0	40.0	13.6	32.7
DeepSeek-R1-Distill-Qwen-7B	38.5	14.0	12.3	12.5	11.4	10.0	10.9	15.7
Qwen2.5-14B No reasoning	55.6	56.3	46.4	46.4	61.1	77.8	31.6	53.6
Qwen2.5-14B CoT	55.6	37.5	42.9	28.6	44.4	55.6	26.3	41.5
DeepSeek-R1-Distill-Qwen-14B	18.8	11.3	10.9	7.2	13.9	24.1	3.9	12.9
Qwen2.5-32B No reasoning	50.0	56.3	44.4	28.0	56.3	66.7	50.0	50.2
Qwen2.5-32B CoT	50.0	25.0	33.3	8.0	43.8	44.4	6.3	30.1
DeepSeek-R1-Distill-Qwen-32B	18.2	2.3	5.7	1.6	3.6	3.8	2.3	5.4

Table 15: Full HSPP results of MBPP+ at varying levels of evaluator reasoning. The numbers in Figure 5 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7B	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.1-8B No reasoning	0	0	0	0	0	0	0	0
Llama-3.1-8B CoT	175	195	191	171	176	156	161	175
DeepSeek-R1-Distill-Llama-8B	1545	1427	1596	1312	1431	1483	1517	1473
Llama-3.3-70B No reasoning	0	0	0	0	0	0	0	0
Llama-3.3-70B CoT	262	223	267	196	240	235	234	237
DeepSeek-R1-Distill-Llama-70B	790	842	948	798	771	815	912	839
Qwen2.5-7B No reasoning	0	0	0	0	0	0	0	0
Qwen2.5-7B CoT	103	95	105	85	101	94	95	97
DeepSeek-R1-Distill-Qwen-7B	821	828	878	784	779	890	893	839
Qwen2.5-14B No reasoning	0	0	0	0	0	0	0	0
Qwen2.5-14B CoT	153	142	158	131	155	151	146	148
DeepSeek-R1-Distill-Qwen-14B	904	921	1084	906	932	934	1028	958
Qwen2.5-32B No reasoning	0	0	0	0	0	0	0	0
Qwen2.5-32B CoT	131	125	132	111	134	128	123	126
DeepSeek-R1-Distill-Qwen-32B	795	768	891	731	728	759	904	796

Table 16: Full HSPP token length results of MATH500. The numbers in Figure 5 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7B	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.1-8B No reasoning	0	0	0	0	0	0	0	0
Llama-3.1-8B CoT	172	183	171	183	187	171	173	177
DeepSeek-R1-Distill-Llama-8B	509	523	533	478	510	503	516	510
Llama-3.3-70B No reasoning	0	0	0	0	0	0	0	0
Llama-3.3-70B CoT	286	303	292	319	304	278	291	296
DeepSeek-R1-Distill-Llama-70B	410	419	417	384	431	400	407	410
Qwen2.5-7B No reasoning	0	0	0	0	0	0	0	0
Qwen2.5-7B CoT	108	111	112	108	108	103	110	108
DeepSeek-R1-Distill-Qwen-7B	467	458	468	441	513	488	453	470
Qwen2.5-14B No reasoning	0	0	0	0	0	0	0	0
Qwen2.5-14B CoT	134	138	134	141	144	136	134	137
DeepSeek-R1-Distill-Qwen-14B	518	516	517	464	525	528	497	509
Qwen2.5-32B No reasoning	0	0	0	0	0	0	0	0
Qwen2.5-32B CoT	126	127	125	131	135	132	136	130
DeepSeek-R1-Distill-Qwen-32B	501	512	501	475	512	489	487	496

Table 17: Full HSPP token length results of MMLU. The numbers in Figure 5 correspond to the average column.

	Evaluatee							
Evaluator	Mistral-7B	Phi-3.5	GPT-3.5-Turbo	GPT-4o	Gemma-2-2B	Llama-3.2-1B	Mistral-Small	Average
Llama-3.1-8B No reasoning	0	0	0	0	0	0	0	0
Llama-3.1-8B CoT	428	388	382	343	396	344	371	379
DeepSeek-R1-Distill-Llama-8B	1435	1310	1336	1168	1429	1312	1305	1328
Llama-3.3-70B No reasoning	0	0	0	0	0	0	0	0
Llama-3.3-70B CoT	265	231	230	229	250	240	226	239
DeepSeek-R1-Distill-Llama-70B	1340	1215	1169	1097	1286	1247	1200	1222
Qwen2.5-7B No reasoning	0	0	0	0	0	0	0	0
Qwen2.5-7B CoT	105	102	101	101	109	101	99	103
DeepSeek-R1-Distill-Qwen-7B	1308	1314	1351	1249	1408	1273	1346	1321
Qwen2.5-14B No reasoning	0	0	0	0	0	0	0	0
Qwen2.5-14B CoT	161	157	169	162	160	155	168	162
DeepSeek-R1-Distill-Qwen-14B	1458	1328	1296	1244	1427	1402	1308	1352
Qwen2.5-32B No reasoning	0	0	0	0	0	0	0	0
Qwen2.5-32B CoT	599	524	556	403	578	513	466	520
DeepSeek-R1-Distill-Qwen-32B	1677	1567	1440	1269	1624	1523	1510	1516

Table 18: Full HSPP token length results of MBPP+. The numbers in Figure 5 correspond to the average column.

C Prompts

You will be given answers from two assistants (Assistant A and Assistant B) to a user prompt. Please act as an impartial judge and evaluate the quality of their answers.

You must output only one of the following choices:

- "A" if Assistant A's answer is better.
- "T" if both answers are relatively the same in quality.
- "B" if Assistant B's answer is better.

For example, if both answers are relatively the same in quality, output "T".

<User Prompt>
{question 1}

<The Start of Assistant A's Answer>
{answer 1}
<The End of Assistant A's Answer>

<The Start of Assistant B's Answer>
{answer 2}
<The End of Assistant B's Answer>

Table 19: Evaluator prompt (no reasoning) for MATH500.

You will be given answers from two assistants (Assistant A and Assistant B) to a user prompt. Please act as an impartial judge and evaluate the quality of their answers.

You must end your response with: "My final verdict is $$...$$.", where $$...$$ must enclose one of the following:

- "A" if Assistant A's answer is better.
- "T" if both answers are relatively the same in quality.
- "B" if Assistant B's answer is better.

For example, if your final verdict is a tie, end your response with: "My final verdict is $$T$$."

<User Prompt>
{question 1}

<The Start of Assistant A's Answer>
{answer 1}
<The End of Assistant A's Answer>

<The Start of Assistant B's Answer>
{answer 2}
<The End of Assistant B's Answer>

Table 20: Evaluator prompt (CoT) for MATH500.

You will be given answers from two assistants (Assistant A and Assistant B) to a user prompt. Please act as an impartial judge and evaluate the quality of their answers.

You must end your response with: "My final verdict is $$...$$.", where $$...$$ must enclose one of the following:

- "A" if Assistant A's answer is better.
- "T" if both answers are relatively the same in quality.
- "B" if Assistant B's answer is better.

For example, if your final verdict is a tie, end your response with: "My final verdict is $$T$$."

<User Prompt>
{question 1}

<The Start of Assistant A's Answer>
{answer 1}
<The End of Assistant A's Answer>

<The Start of Assistant B's Answer>
{answer 2}
<The End of Assistant B's Answer>

<think>

Table 21: Evaluator prompt (long CoT) for MATH500.

You will be given answers from two assistants (Assistant A and Assistant B) to a user prompt. Please act as an impartial judge and evaluate the quality of their answers.

You must output only one of the following choices:

- "A" if Assistant A's answer is better.
- "T" if both answers are relatively the same in quality.
- "B" if Assistant B's answer is better.

For example, if both answers are relatively the same in quality, output "T".

<User Prompt>
{question 1}

<The Start of Assistant A's Answer>
{answer 1}
<The End of Assistant A's Answer>

<The Start of Assistant B's Answer>
{answer 2}
<The End of Assistant B's Answer>

Table 22: Evaluator prompt (no reasoning) for MMLU.

You are given answers provided by two assistants, Assistant A and Assistant B, to the following user prompt.

<User Prompt>
{question 1}

<The Start of Assistant A's Answer>
{answer 1}
<The End of Assistant A's Answer>

<The Start of Assistant B's Answer>
{answer 2}
<The End of Assistant B's Answer>

Please act as an impartial judge and evaluate the quality of their answers.

You must end your response with: "My final verdict is $$...$$.", where $$...$$ must enclose one of the following:

- "A" if Assistant A's answer is better.
- "T" if both answers are relatively the same in quality.
- "B" if Assistant B's answer is better.

For example, if your final verdict is a tie, end your response with: "My final verdict is $$T$$."

Table 23: Evaluator prompt (CoT) for MMLU.

You are given answers provided by two assistants, Assistant A and Assistant B, to the following user prompt.

<User Prompt>
{question 1}

<The Start of Assistant A's Answer>
{answer 1}
<The End of Assistant A's Answer>

<The Start of Assistant B's Answer>
{answer 2}
<The End of Assistant B's Answer>

Please act as an impartial judge and evaluate the quality of their answers.

You must end your response with: "My final verdict is $$...$$.", where $$...$$ must enclose one of the following:

- "A" if Assistant A's answer is better.
- "T" if both answers are relatively the same in quality.
- "B" if Assistant B's answer is better.

For example, if your final verdict is a tie, end your response with: "My final verdict is $$T$$."

<think>

Table 24: Evaluator prompt (long CoT) for MMLU.

Please act as an impartial judge and assess the accuracy of the given Python functions. Provide your evaluation by outputting only one of the following

- "A" if Assistant A's function is more accurate.
- "T" if both functions are similarly accurate.
- "B" if Assistant B's function is more accurate.

Base your judgment on your understanding of the task described in the user prompt and whether the functions successfully pass the provided test cases.

<User Prompt>
{question 1}

<The Start of Assistant A's Function>
{answer 1}
<The End of Assistant A's Function>

<The Start of Assistant B's Function>
{answer 2}
<The End of Assistant B's Function>

Table 25: Evaluator prompt (no reasoning) for MBPP+.

Please act as an impartial judge and assess the accuracy of the given Python functions. You must end your response with: "My final verdict is $$...$$.", where $$...$$ must enclose one of the following:

- "A" if Assistant A's function is more accurate.
- "T" if both functions are similarly accurate.
- "B" if Assistant B's function is more accurate.

For example, if your final verdict is a tie, end your response with: "My final verdict is $$T$$."
Base your judgment on your understanding of the task described in the user prompt and whether the functions successfully pass the provided test cases.

<User Prompt>
{question 1}

<The Start of Assistant A's Function>
{answer 1}
<The End of Assistant A's Function>

<The Start of Assistant B's Function>
{answer 2}
<The End of Assistant B's Function>

Table 26: Evaluator prompt (CoT) for MBPP+.

Please act as an impartial judge and assess the accuracy of the given Python functions. You must end your response with: "My final verdict is $$...$$.", where $$...$$ must enclose one of the following:

- "A" if Assistant A's function is more accurate.
- "T" if both functions are similarly accurate.
- "B" if Assistant B's function is more accurate.

For example, if your final verdict is a tie, end your response with: "My final verdict is $$T$$." Base your judgment on your understanding of the task described in the user prompt and whether the functions successfully pass the provided test cases.

<User Prompt>
{question 1}

<The Start of Assistant A's Function>
{answer 1}
<The End of Assistant A's Function>

<The Start of Assistant B's Function>
{answer 2}
<The End of Assistant B's Function>

<think>

Table 27: Evaluator prompt (long CoT) for MBPP+.

Answer the given problem by providing your solution. You must use \boxed{...} to enclose the final answer.

Problem:
{question}

Solution:

Table 28: Generator prompt (instruction-tuned model) for MATH500.

Answer the given problem by providing your solution. You must use \boxed{...} to enclose the final answer.

Problem:
{question}

Solution:
<think>

Table 29: Generator prompt (reasoning model) for MATH500.

```
Answer the given multiple-choice problem. If you cannot determine the correct answer, take your best guess.
You must end your response with "The final answer is $$...$$.", where $$...$$ must only enclose the label of your final answer. For example, if your final answer is K, then write "The final answer is $$K$$.".
Problem:
{question}

Solution:
```

Table 30: Generator prompt (instruction-tuned model) for MMLU.

```
Answer the given multiple-choice problem. If you cannot determine the correct answer, make your best guess.
You must use $$...$$ to enclose the label of your final answer. For example, if your final answer is K, your response should contain $$K$$.
Problem:
{question}

Solution:
<think>
```

Table 31: Generator prompt (reasoning model) for MMLU.

```
You are a Python programmer.

{question}
```

Table 32: Generator prompt (instruction-tuned model) for MBPP+.

```
You are a Python programmer.

{question}
<think>
```

Table 33: Generator prompt (reasoning model) for MBPP+.