JuStRank: Benchmarking LLM Judges for System Ranking

Ariel Gera, Odellia Boni, Yotam Perlitz,
Roy Bar-Haim, Lilach Eden and Asaf Yehudai
IBM Research

Abstract

Given the rapid progress of generative AI, there is a pressing need to systematically compare and choose between the numerous models and configurations available. The scale and versatility of such evaluations make the use of LLM-based judges a compelling solution for this challenge. Crucially, this approach requires first to validate the quality of the LLM judge itself. Previous work has focused on *instance-based* assessment of LLM judges, where a judge is evaluated over a set of responses, or response pairs, while being agnostic to their source systems. We argue that this setting overlooks critical factors affecting system-level ranking, such as a judge's positive or negative bias towards certain systems. To address this gap, we conduct the first large-scale study of LLM judges as *system rankers*. System scores are generated by aggregating judgment scores over multiple system outputs, and the judge's quality is assessed by comparing the resulting system ranking to a human-based ranking. Beyond overall judge assessment, our analysis provides a fine-grained characterization of judge behavior, including their *decisiveness* and *bias*.

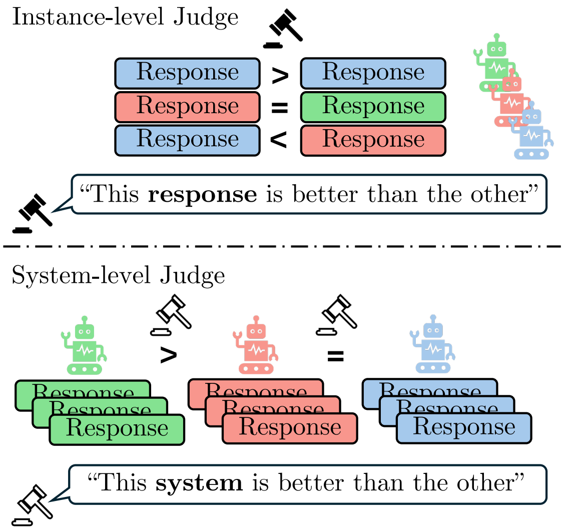


Figure 1: **Instance and system level judges make different calls:** An *instance-level* judge (top) is used to make decisions about the quality of individual responses (which may be produced by different systems). A *system-level judge* (bottom) is used to make decisions about the overall quality of systems. For clarity, in this illustration, we focus on pairwise decisions.

1 Introduction

The evaluation of Large Language Models (LLMs) is rapidly adopting the LLM-as-a-judge paradigm (Zheng et al., 2023), where automatic evaluations with LLMs complement the use of human annotators, or even replace them altogether. LLM-based judges are increasingly relied upon to conclude which models exhibit superior performance, whether novel training and inference approaches are beneficial, and ultimately which LLM configurations offer a better value proposition to users.

Since relying on an inaccurate judge will likely result in sub-optimal decisions, this trend lends an urgency to evaluating the performance of the LLM judges themselves. Indeed, recent works attempt to benchmark judging capabilities, compiling leaderboards of judge performance (Lambert et al., 2024; Tan et al., 2024) as well as analyzing their sensitivities and biases (Wang et al., 2023; Wei et al., 2024; Bavaresco et al., 2024; Feuer et al., 2024; Liu et al., 2024b; Xu et al., 2024; Ye et al., 2024).

These works all focus on the *instance-level performance* of judges. A "good" instance-level judge is expected to make a correct judgment about each response, regardless of the system generating it. For example, given a specific pair of responses, the judge may be asked to determine which one is better (Figure 1, top). This approach is very much in line with prevailing paradigms for model alignment (e.g., RLHF, DPO; Lee et al., 2024b) and synthetic data generation (Yehudai et al., 2024); these often rely on LLM judges and reward models for making

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 682–712
July 27 - August 1, 2025

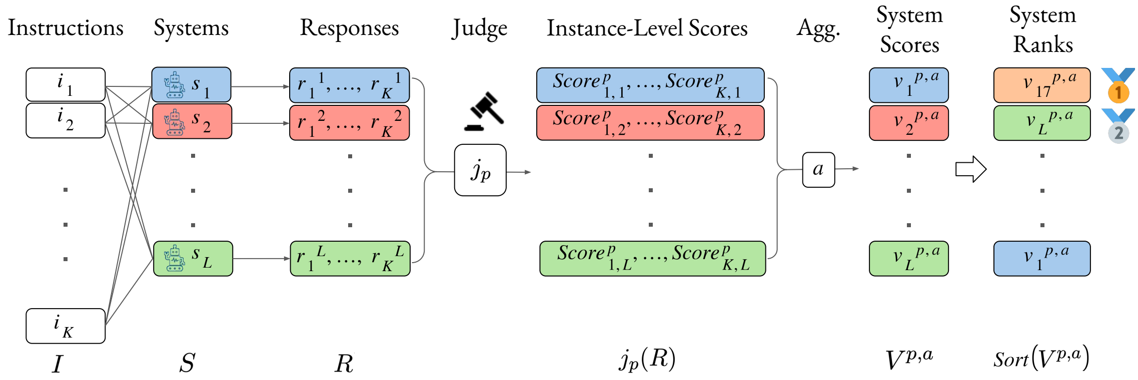


Figure 2: **System-level judge pipeline.** Schematic of our data generation pipeline for judge system rankings.

instance-level pairwise decisions on the quality of individual responses.

Although judges are **evaluated** based on their instance-level performance, very commonly they are actually **used** for making system-level decisions; namely, to compare and rank different models or different configurations (Figure 1, bottom). Crucially, even very good instance-level capabilities do not guarantee accurate model ranking; and at the same time, mediocre performance on instances could still yield a very accurate overall ranking (Dorner et al., 2024, §2). Thus, the *system-level performance* of judges – that is, to what degree they can correctly decide between candidate systems, and produce accurate model performance rankings – remains largely an open question. Furthermore, system-level evaluations can unveil an entire range of under-explored judge qualities, such as being biased towards certain models or making un-calibrated model preference judgments.

In this work we aim to address this gap, and characterize the system-level evaluation capabilities and behaviors of LLM-based judges. To this end, we introduce a novel judge benchmark – JuStRank *(Judges for System Ranking)*. JuStRank compares judges by their ability to correctly rank models, based on agreement with a ground-truth model ranking. JuStRank encompasses a collection of 48 state-of-the-art judges, including both general-purpose LLMs and reward models. Our large-scale benchmark and analysis allow us to explore the performance and behavior of judges as system rankers.

Our contributions are as follows:

1. We introduce JuStRank, the first large-scale benchmark of judges for ranking target systems.

2. We quantify the tendency of a judge to exhibit *system bias*, where some models are judged "unfairly" (§6.2).

3. We reveal an emergent quality of a system-level judge, its *decisiveness* factor; decisive judges consistently amplify the gap between strong and weak target systems (§6.1).

4. To facilitate further research into judge behavior, we release our data[1], comprising 1.5M judgment scores given by LLMs and reward models.

2 The Gap in Judge Benchmarking

In this section, we outline why existing estimations of judge performance are insufficient to decide which judge is best at choosing between target systems (Figure 1, bottom).

At present, users looking for a judge for ranking models, will likely choose it according to the available instance-level judge benchmarks. Yet, from a theoretical standpoint instance-level judge performance does not directly correspond to system-level judge performance (Dorner et al., 2024).

More specifically, instance-level judge evaluations focus on *how many* errors the judge makes, and do not address the *distribution* of these errors across systems.

For system-level judge evaluation, however, the error distribution plays a key role, as judge errors may distribute unevenly across systems, impacting their induced ranking (Dorner et al., 2024; von Däniken et al., 2024). For example, a judge may exhibit an unjustifiable preference (positive bias) for responses from a particular system A. Thus, this judge will tend to give A an incorrect ranking, even if it makes very few mistakes on responses from other systems (i.e., has an overall high instance-level accuracy). Hence, a more uniform distribution of errors – reflecting less biased judgment – is a desirable quality for system-level judges, and one that may lead to a more accurate ranking.

[1] JuStRank Judge Scores data

Drawing on this observation, our goal here is to construct a system-level benchmark for judges. As a benchmark tailored for system-level evaluation, it will enable reliably estimating a judge's ability to rank systems; moreover, our ranking-oriented analysis can shed light on judge behaviors and biases, as they occur in real-world data.

3 Task Formulation

In this work we study the use of LLM-based judges for determining the relative quality of systems[2], over a given set of user instructions (prompts).

Formally, we begin with a set of L systems $\mathbf{S} = \{s_l\}_{l=1}^{L}$, and K user instructions $\mathbf{I} = \{i_k\}_{k=1}^{K}$. Each system produces a response for each such user instruction, denoted as $R = \{r_k^l\}_{k,l=1,1}^{k,l=K,L}$, such that $s_l(i_k) = r_k^l$ (see Figure 2).

Judges $\mathbf{J} = \{j_p\}_{p=1}^{P}$ map a pair of instruction i_k, and system response r_k^l to a scalar score that estimates the quality of the response. Each judge has a specific *realization* for performing this score mapping[3], of the form: $j_p(i_k, r_k^l) = Score_{k,l}^p$. Once a judge j_p scores all $K \times L$ responses, we can define a scores matrix $j_p(R) \in \mathbb{R}^{K \times L}$ where $j_p(R)_{k,l} = Score_{k,l}^p$.

In order to quantify system-level quality, we must apply an *aggregation method*, $a \in A = \{a\colon \mathbb{R}^{K\times L} \longrightarrow \mathbb{R}^L\}$. The aggregation method a maps a scores matrix $j_p(R)$ to a system-level vector $V^{p,a} \in \mathbb{R}^L$ where each entry, $V_l^{p,a}$, is a single overall quality score for system s_l by judge j_p. In turn, ordering the system scores in $V^{p,a}$ induces a ranking over the systems set $\mathbf{S}$.

We test the performance of judge j_p as a ranker by checking the correlation between the ranking induced by $V^{p,a}$ and a golden ranking for $\mathbf{S}$.

4 Experimental setup

To explore judge performance and behavior, we utilize responses from multiple systems (§4.1) and run reward model judges (§4.2.1) and LLM judges (§4.2.2) over these responses. To obtain system rankings, we experiment with different aggregation methods (§4.3) over the judge scores. Finally, the resulting rankings are compared against a gold system ranking, taken from a separate dataset (§4.4).

4.1 System Responses Data

We utilize the Arena Hard v0.1 dataset (Li et al., 2024) for a diverse set of instructions and system responses. The dataset uses a curated set of $K = 500$ challenging instructions, I. As of September 2024, it includes responses from $L = 63$ systems, S, totaling about 32K pairs of instructions and their associated system responses, R.

4.2 Generating Judgments

For every judge realization, j_p, we generate a judgment scores matrix, $j_p(R)$, over R. In total, we examine 48 judge realizations, yielding a total of 1.5M individual judge scores (63 systems $\times$ 500 instances $\times$ 48 judge realizations).

4.2.1 Reward Models

We run multiple reward models over R. While their exact architectures vary, reward models generally produce a scalar quality score for a given pair of an instruction and a system response.

We utilize the following reward models: ArmoRM-Llama3-8B-v0.1 (Wang et al., 2024), Eurus-RM-7b (Yuan et al., 2024), InternLM2-7b-reward, InternLM2-20b-reward (Cai et al., 2024), Skywork-Reward-Llama-3.1-8B-v0.2 (Liu et al., 2024a), Llama-3-OffsetBias-RM-8B (Park et al., 2024), GRM-Llama3.2-3B-ft (Yang et al., 2024), URM-LLaMa-3.1-8B (Lou et al., 2024).

4.2.2 LLM Judge Realizations

Unlike dedicated reward models that produce a single score, generative LLMs can be prompted to judge in multiple ways. Thus, for every LLM we examine several judge realizations.

Absolute judgment - Numeric score *(Numeric)* The LLM judge is given an instruction and system response, and is asked to provide a quality score for the response between 0 and 100.

Absolute judgment - Textual score (*Likert*) The judge provides a quality score of the response on a Likert (Likert, 1932) scale with 5 labels: *[Very Bad, Bad, Mediocre, Good, Very Good]*. We then convert the textual judgments to scores in $[1-5]$.

Absolute judgment - Token probablities (*TokenProbs*) The task is framed to the judge as a yes/no question: *Is this a good response?*. We then extract the top log-probabilities for the first

[2]Henceforth, we will use the term *System* to refer to a target model or pipeline that performs a task, and *Judge* for one that is asked to score (or compare) the quality of such systems. Generative LLMs can act as both systems and judges.

[3]We note that some realizations, such as the comparative realization in §4.2.2, may incorporate a separate set of responses to perform the judgment.

Judge Model	Realization	Aggregation	Agreement (τ) with Gold Ranking
Qwen2.5-72B-Instruct	Likert	Win-Rate	.83
URM-LLaMa-3.1-8B	Reward	Mean	.82
GPT-4o-2024-11-20	Anchor	Mean	.82
Llama-3-1-405b-instruct-fp8	Numeric	Mean	.81
Mistral-large-instruct-2407	Likert	BT	.81
GPT-4o-mini-2024-07-18	Numeric	Win-Rate	.81
ArmoRM-Llama3-8B-v0.1	Reward	Mean	.80
Llama-3-1-70b-instruct	Numeric	Win-Rate	.80
Skywork-Llama-3.1-8B-v0.2	Reward	Mean	.79
Llama-3.1-8B-Instruct	TokenProbs	Mean	.78

Table 1: **Top 10 judges by ranking performance**. Judges are sorted by the Kendall's Tau correlation between their overall system ranking and the gold ranking from Chatbot Arena (§4.4). For every judge model, only the best-performing realization and aggregation method is shown. For the full results, refer to Appendix Table 2.

generated token, and specifically look at the probabilities for the tokens *yes* or *no*. The judgment score $[0.0 - 1.0]$ is the sum of probabilities for *yes* divided by the sum of probabilities for *yes* and *no*.

Comparative judgment - Anchor model (*Anchor*) Here the judgment task is comparative, i.e., the judge is asked to state a preference between two responses rather than an absolute quality judgment of a given response. Conducting paired comparisons between a system and all other systems is unfeasible; thus, we follow Li et al. (2024) and use the responses of *GPT-4-0314* as *anchors* to which the responses of other systems are compared. Given an anchor response and a system response, we ask the judge which one it prefers. The output is then converted to scores in $[-2, +2]$ (where 0 indicates a tie, and $+1$ / $+2$ indicate slight/strong preference for the system response over the anchor response, respectively).

In total, we collect judgments from 10 LLMs and 4 realizations, yielding 40 LLM judges. Prompts for all realizations are provided in Appendix G.

We use the following generative LLM judges: Llama-3.1-405B-Instruct (Dubey et al., 2024), Llama-3.1-70B-Instruct, Llama-3.1-8B-Instruct, Llama-3-70B-Instruct, Mixtral-8x22B-Instruct-v0.1, Mixtral-8x7B-Instruct-v0.1 (Jiang et al., 2024), Mistral-Large-Instruct-2407, Qwen2.5-72B-Instruct, GPT-4o and GPT-4o-mini.

4.3 Aggregations

Given the raw judgment scores of each judge, $j_p(R)$, there are multiple ways to construct a *ranking* of the 63 target systems. We calculate rankings using **Win-rate** aggregation, **Mean** aggregation, **Median** aggregation, and **BT** (Bradley-Terry) aggregation. Details are provided in Appendix B.

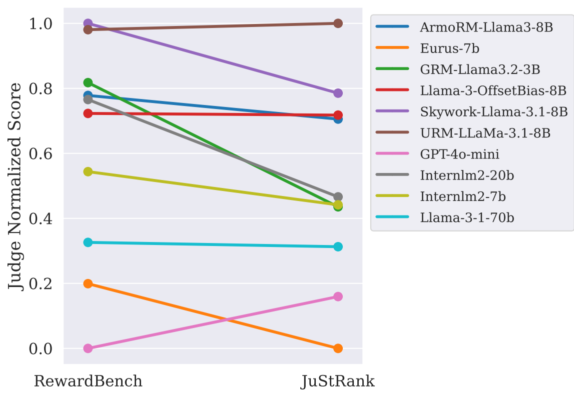


Figure 3: **Comparison to RewardBench**. The plot depicts the relative performance of judges present in both JuStRank and RewardBench (Lambert et al., 2024). For comparison, we perform Min-Max normalization over the judge performance scores (*accuracy* for RewardBench, *Kendall's Tau* for our results). Results shown are for the BT aggregation method; the LLM judges use the *Anchor* realization, which is closest to the setting in RewardBench. Plots for the different RewardBench subsets are shown in Appendix Figure 8.

4.4 Gold Ranking - Chatbot Arena Battles

Human preference data from Chatbot Arena (Zheng et al., 2023) serve as our ground-truth reference for the relative quality of systems. Chatbot Arena relies on human-annotated "battles" between system responses to produce a system

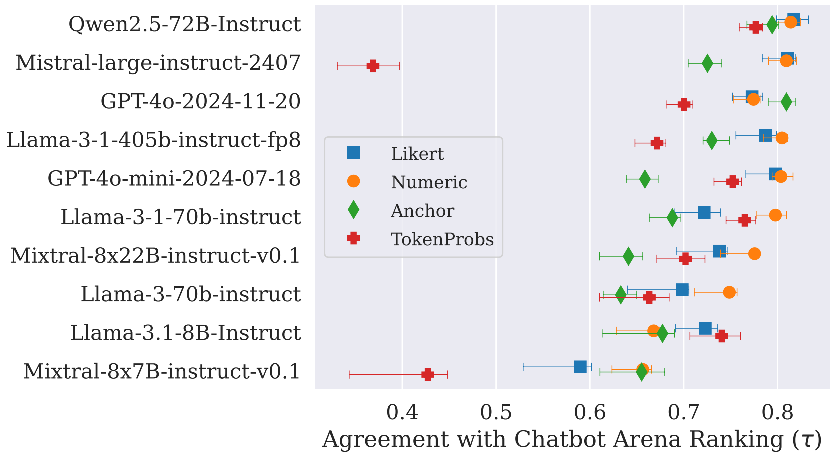


Figure 4: **LLM judge realizations**. Kendall's Tau correlations ($\pm 95\%$ bootstrapping CI) between the system rankings produced by various LLM judge realizations (§4.2.2) and the gold system ranking from Chatbot Arena. The plot depicts results for the BT aggregation method; for the full results, refer to App. Table 2.

ranking. We use the *English Hard Prompts* subset[4] of their data. We chose this subset as its distribution of user instructions has been shown (Li et al., 2024) to match that of our system response data (§4.1). We extract the data and ranking following the official code (see Appendix C).

Given a system ranking produced by a judge, we quantify judge performance via the correlation between its ranking and the reference ranking from Chatbot Arena. Simply put, we assume that a ranking given by a good automated judge would have a high agreement with the ranking compiled from human judgments.

5 JuStRank - Judge Performance Results

Table 1 depicts the 10 top-performing judges on JuStRank, based on their ranking agreement (τ) with the ground-truth human ranking from Chatbot Arena. For each judge model, the best-performing realization and aggregation method is shown.

As seen in the table, there are both LLMs and reward models that reach decent agreement with the gold ranking. Moreover, several 8B-parameter reward models are on par with much larger LLMs on the task of system ranking. Thus, we see that reward models, which are explicitly trained to make instance-level decisions between pairs of responses, can excel at the system-level ranking task as well.

Note that an identical correlation score with the ground-truth ranking does not indicate that the judges produce the *same* ranking; rather, each judge has a different pattern of agreement with the ground-truth. Correlations among the judges themselves are shown in App. Fig. 9.

Comparison to Instance-Level Performance In Figure 3 we compare our system-level judge leaderboard to the instance-level benchmark RewardBench (Lambert et al., 2024). The results demonstrate that better instance-level judges are not always better system rankers, highlighting the discrepancy between the two tasks. Thus, JuStRank offers a novel perspective on judge ability. However, there may be additional factors at play as well. For LLM judges, we use a slightly different realization from the comparative prompts used for RewardBench. Moreover, since creators of reward models aim to do well on RewardBench, it is possible that some newer reward models are slightly overfitted to this test distribution.

5.1 Effects of LLM Realizations

Figure 4 depicts the performance of the LLM judge models by their realization (§4.2.2). The plot demonstrates that the choice of realization has a considerable effect on the system ranking quality; this appears to be nearly as important as the identity of the LLM used. We confirm this finding using statistical variance analysis (Appendix D).

Many works recommend asking LLMs for comparative rather than absolute judgments (Zheng et al., 2023). However, in our experiments the comparative realization (*Anchor*) exhibits lower performance, with the notable exception of GPT-4o. The best realizations overall were *Numeric* and *Likert*, where the judge is asked to provide a ver-

[4] Chatbot Arena Hard Prompts

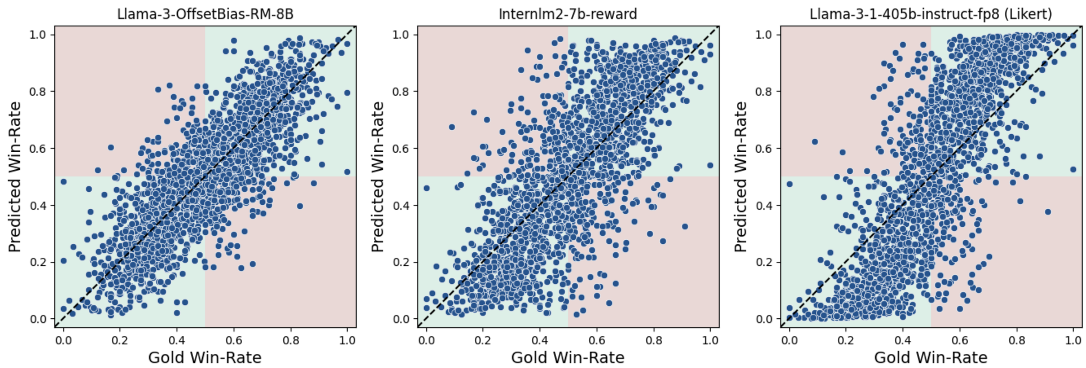


Figure 5: **Predicted pairwise win-rates**. Each point represents a win-rate between a pair of systems $WR(s_a, s_b)$ (App. E). The x-axis denotes the gold win-rate from Chatbot Arena, and the y-axis denotes the predicted win-rate as derived from the judge scores. The diagonal marks an exact match between the predicted and gold win-rate; the quadrants signify whether the predicted winning system is the same (green) or different (red) from the gold winning system for this pair. Note that every pair is represented twice (e.g., $WR(s_a, s_b) = 0.2$, $WR(s_b, s_a) = 0.8$).

balized quality score. This is in line with findings from Tian et al. (2023), who report better calibration with verbalized LLM confidence scores. The higher performance for both *Numeric* and *Likert* realizations – compared to *Anchor* and *TokenProbs* – is statistically significant (App. D).

We also note that each realization induces a characteristic distribution of judge scores, D^p, such that $Score^p_{k,l} \sim D^p$. Notably, the LLM judges tend to produce particular score values more often than others. Refer to Appendix A for more details.

6 Judge Behavior

Next, we explore more fine-grained judge behaviors, beyond the bottom-line system rankings.

To that end, we focus on the judgment task of pairwise system preference, as this is the foundation of system ranking tasks. As in §5, our aim is to gain an understanding of judge performance and characteristics, by comparing judge behavior on pairwise system preference to ground-truth data.

Pairwise Win-Rates For every judge j_p, and for every pair of systems (s_a, s_b), the win-rate of s_a, denoted by $WR^p(s_a, s_b)$, is the number of instances where it received a higher score than s_b, divided by the number of non-tied instances (cf. Appendix E). Thus, we calculate the pairwise win-rate for each system pair according to each judge. Note that the win-rates are calculated on the scores matrix $j_p(R)$, i.e., before applying an aggregation method.

Gold Win-Rates Similarly, we extract gold pairwise win-rates, WR^g, from Chatbot Arena (App. C). 59 systems appear both in our response data (§4.1) and in Chatbot Arena; in total, we have both judge and gold data for 968 head-to-head comparisons between pairs of systems.

6.1 Some Judges are Particularly Decisive

Figure 5 depicts the relationship between predicted win-rates and gold win-rates for several judges. The quadrants in the figure indicate whether the judge's pairwise preference decision is aligned with the gold preference. As can be expected, the judge predictions in Figure 5 are often centered around the ground-truth win-rates determined by humans. But strikingly, some judges exhibit unique prediction patterns, yielding win-rates that are consistently closer to the extremes (0.0 / 1.0) compared to the human data. For instance, for pairs with a ground-truth win-rate of ~0.8, the predicted win-rate in the judgments of Llama-405B (Fig. 5, right) tends to exceed 0.9. Put simply, when faced with a response from a strong system, the judge is very likely to prefer it over the response of a less capable system, even where human judges are less decisive.

This sigmoidal win-rate prediction pattern resembles behaviors previously described for classifier calibration (Silva Filho et al., 2023), where classifiers may exhibit "overconfidence" in their predicted probabilities.[5] Thus, following Kull et al. (2017), we quantify judges' decisive (overconfident) behavior by fitting the cumulative beta distribution function to the win-rate prediction plots. This enables describing judge prediction behav-

[5]Note, however, that the behavior in our case does not reflect judge probability scores, but rather the empirical ratio of instances where the responses $\{r^l_k\}_{l=1}^{l=L}$ of a system k are preferred over those of another system.

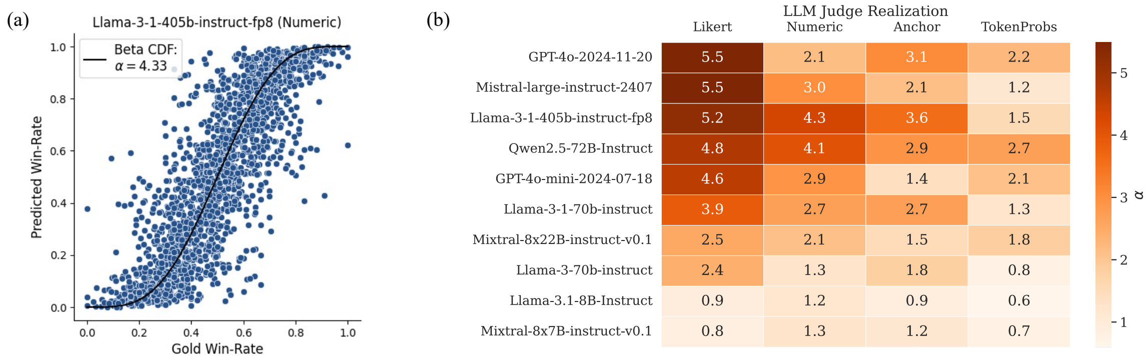


Figure 6: **Beta distribution fit of pairwise win-rates**. (a): *Judge beta fit example*. Each point represents the win-rate between a pair of systems, $WR(s_a, s_b)$; the curve and α value describe a fit to the beta distribution (App. F). Plots for all judges are in App. Fig. 11. (b): *Decisiveness by judge realization*. Cell values denote the decisiveness behaviors of different LLM judge realizations, as described by the α value for their win-rate distribution.

ior in terms of a single fit value $\alpha = \beta$, where $\alpha \in [0, \infty]$, a value of $\alpha = 1$ represents no over- or under-decisiveness, and larger values represent more decisive behavior (refer to Appendix F for details). Figure 6a and App. Fig. 11 depict the beta curve fit for win-rates of various judges.

Figure 6b compares judge realizations in terms of their decisiveness behavior. We see that LLM judges are usually more decisive when directly asked to provide a quality score, and in particular a textual one (*Likert*); in contrast, the realization that relies on token probabilities (*TokenProbs*) does not give rise to such a pattern, and can even result in judge "indecision" (i.e., $\alpha < 1$).

This pattern can be explained from two directions. First, the human judgments (§4.4) were collected from multiple individuals, who likely have differing preferences; this may introduce some noise that could lead to *less extreme win-rates in the gold data*. The other factor is the judges, who may rely on certain heuristics to identify responses from strong systems (Feuer et al., 2024), leading to *more extreme win-rates in the judge data*. While the variance between judges (Fig. 6b) supports the latter, we cannot determine this conclusively.

In practical terms, extreme win-rates can be beneficial to users, as they increase the likelihood of a correct system preference decision given a smaller set of responses (see Ashury Tahan et al., 2024).

6.2 Bias Towards Specific Systems

A major concern when using judges for system preference is *judge bias* – a judge may treat a specific system "unfairly", by consistently judging its responses too favorably or too harshly (see Von Däniken et al., 2024).

We define the bias $B^p_{s_a}$ of judge j_p towards system s_a by the expectation over the differences between the predicted and gold win-rates, over all systems that s_a interacts with. Formally, $B^p_{s_a} = \mathbb{E}_{s_b \in S}(WR^p(s_a, s_b) - WR^g(s_a, s_b))$.[6] In other words, if according to j_p the win-rates of system s_a are (on average) higher than those in the human data, we will say that j_p exhibits *positive bias* towards it; and if they are lower than the ground-truth, j_p would be said to exhibit *negative bias*.

Note that the decisiveness behavior in §6.1 directly entails a general bias pattern in some judges – namely, a positive bias towards strong systems, and a negative bias towards weak ones. Thus, we calculate a decisiveness-corrected bias, ${B'}^{p}_{s_a}$, where the gold win-rate WR^g is replaced by $WR^{g'_p}$, i.e., the predicted value for the gold win-rate on the beta distribution fit for judge j_p (App. F).

We observe some consistent trends of system-specific bias that are common across judges. Figure 7 depicts systems for which there is high bias across judges. For instance, most judges exhibit a strong positive bias towards Athene-70B, to the extent that it is often ranked by them as the #1 system. In contrast, GPT-4-0613, which is 27th in the gold ranking, receives negative bias, resulting in a median rank of 38 among the judges.

We also ask whether LLM judges exhibit *self-bias* (Xu et al., 2024; Panickssery et al., 2024), i.e., bias towards the system that uses the same underlying LLM. While we find some instances of

[6] Our formulation of bias aims to reflects the practical impact of the judge bias on system preference. This is in contrast to the Favi-Score metric proposed by Von Däniken et al. (2024), which is decoupled from the overall accuracy of preference decisions.

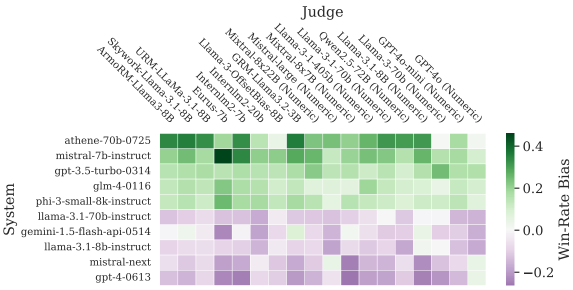


Figure 7: **System-specific judge biases**. The plot depicts win-rate biases of judges towards specific systems, with respect to the ground-truth win-rates from Chatbot Arena (after correction for the beta distribution fit of each judge). This plot portrays select systems with high bias; the full heat map, including all judge realizations and all systems, is shown in App. Fig. 10b.

self-bias, this is not a consistent effect across judge realizations (App. Table 3).

To quantify the overall propensity of a judge for bias, we measure the standard deviation of its bias over all systems, $\delta = \sigma_{s \in S}(B'^p)$. The bias measure for each judge is presented in App. Table 4.

6.3 Characterizing Judge Behaviors

We have shown that beyond their overall ranking capability (§5), judges exhibit distinct traits in their system-level judgments – in particular, they show different levels of *decisiveness* (§6.1), and overall propensities for *bias* (§6.2). Interestingly, each of these traits (cf. App. Table 4) is correlated to the ranking quality τ, with $r = 0.55$ for the α decisiveness measure, and $r = -0.56$ for the bias propensity δ. At the same time, these marked traits are – by design – uncorrelated with each other ($r = -0.07$ between α and δ). Thus, our analyses reveal global system-level judge traits, ones that remain hidden when assessing judges from an instance-level perspective.

7 Related Work

Applying and assessing automatic metrics for system-level evaluation has been studied for decades, in particular for natural language generation tasks (Reiter and Belz, 2009; Louis and Nenkova, 2013; Deutsch et al., 2022). In the context of LLM-based judges, however, system-level evaluation is still under-explored.

Prior works on LLM-based judges have opted for an instance-level evaluation approach, curating benchmarks of responses with ground-truth quality annotations in order to evaluate judge performance. Most prominently, RewardBench (Lambert et al., 2024) compares dozens of judges (including reward models, generative LLMs, and classifiers) on the task of deciding between pairs of outputs. RewardBench aims to identify the most suitable judges for model alignment, e.g., for use in RLHF; in contrast, our work measures judges in terms of their ability to compare the performance of candidate systems. Another recent instance-level benchmark, JudgeBench (Tan et al., 2024), focuses on curating challenging response pairs where the judge must discern subtle errors.

Multiple works are dedicated to analyzing various biases (Ye et al., 2024) and undesirable behaviors exhibited by judges. These include positional bias (Wang et al., 2023), verbosity bias (Saito et al., 2023; Chen et al., 2024) and self-bias (Xu et al., 2024), as well as sensitivity to prompts (Wei et al., 2024), source datasets (Bavaresco et al., 2024), epistemic markers (Lee et al., 2024a) and style (Feuer et al., 2024; Liu et al., 2024b).

Several popular benchmarks rely on LLM judges to produce leaderboards of state-of-the-art systems. Such benchmarks – e.g., Arena Hard (Li et al., 2024) and AlpacaEval (Dubois et al., 2024) – do perform a system-level validation of their resulting leaderboards against other benchmark rankings (see Perlitz et al., 2024). However, such efforts are limited to validating the particular dataset and judge setup chosen for the benchmark (usually with GPT-4 as the judge), rather than comparing and analyzing the performance of different judge models and implementations. Thakur et al., 2024 conduct a task-specific system-level evaluation of judges, over the TriviaQA (Joshi et al., 2017) dataset. Compared to their work, the present study is on a larger scale and offers novel metrics and analyses on system-level judge behaviors.

8 Discussion

The usage of LLM-based judges is continually expanding. Moreover, many research papers – proposing novel architectures, algorithms and training methods – rely heavily on system-level evaluations using judges as evidence for the utility of their approach. But without evaluating the judges on such system-level tasks, how can one know whether to trust such evaluations, and their conclusions?

We are the first to investigate on a large scale the performance of LLM-based judges on the system ranking task. Our resulting benchmark, JuStRank, will assist users and researchers in choosing the judge best suited for their needs.

Choosing a judge requires many fine-grained decisions. A user can decide which reward model or LLM to use as the judge; opt for relative judgments or absolute scores; try various prompts; apply different aggregations to compile a ranking, etc. Furthermore, these decisions may interact in non-trivial ways (e.g., the distribution of scores a judge tends to output can dictate which aggregations will work well). Indeed, our findings confirm that such decisions substantially affect system-level judgments (§5), and thus are quite likely to change the model selection of an end user, or flip the conclusions of an NLP research paper.

Our system-level approach has multiple additional benefits. First, it forces the evaluation of judges to be representative with respect to *the distribution of systems that generate the responses*. In existing instance-level benchmarks this factor is not taken into account, and likely results in less accurate judge evaluations. Second, it affords a new perspective on what it means for a judge to be biased; on the one hand, we discover some decisiveness trends (§6.1) that may actually be useful for making correct preference decisions, and increasing the separability between systems; and on the other, we report some problematic biases that directly distort the judgment of particular systems (§6.2). An important avenue for future work is to connect our findings here to the existing literature on judge biases (Ye et al., 2024), and understand to what extent both of these behaviors stem from particular LLM style attributes (Feuer et al., 2024).

Given this vast and complex space, our work is admittedly only a first step in understanding the behavior of judges for ranking and selecting LLMs. We release our raw judgment scores data, and encourage the community to explore these issues further: for instance, by training dedicated system-level judges, exploring judge ensembles, or studying other aggregation approaches. We believe that `JuStRank` can facilitate such research directions, as it can be easily extended to new judges without requiring additional human annotations.

Our hope is that both practitioners and researchers can benefit from `JuStRank`, by making more informed choices of judges for their needs.

9 Conclusion

In this work we conducted the first comprehensive evaluation of system ranking by LLM judges. We tested a wide array of judges, including reward models and different realizations of generative LLMs, over a large collection of systems.

We collected system responses over a diverse set of instructions. The judges scored each response, and we compiled a ranking by aggregating the judgments over all responses. Then, the quality of the judge's system ranking was compared to a human ranking, producing the `JuStRank` leaderboard.

`JuStRank` allows users to pick judges that are better aligned with the goal of choosing between different models and configurations. `JuStRank` demonstrates that judge ranking abilities are not directly tied to LLM size or overall quality, and that some dedicated reward models are on par with leading LLM judges. Moreover, our analysis reveals emergent judge traits – *decisiveness* and *bias* – that are strongly correlated with their ranking ability.

Limitations

The gold reference data – the *English Hard Prompts* subset of Chatbot Arena – does not include user instructions or responses. Hence, we collect judgment data over Arena Hard, which contains a large set of instructions and responses. This raises some questions regarding our ability to directly compare the LLM judges and human judges. However, given that Arena Hard was designed to match the distribution of user instructions in *English Hard Prompts* (see Li et al., 2024), we assume that these datasets are sufficiently similar.

Our analyses of LLM judge realizations are, by necessity, limited to the specific realization prompts that we used. Several studies show that LLMs (Mizrahi et al., 2024) as well as LLM judges (Wei et al., 2024) are brittle with respect to prompt phrasing, and hence this may have had an impact on the results. In addition, there can be some variations in judge responses depending on the exact API and inference implementation used.

As in multiple other works, here we treat human preference as a single concept. In practice, however, preference is inherently subjective, and is composed of numerous dimensions (e.g., helpfulness, safety, style, coherence etc.). For instance, one individual may prefer succinct model responses while another would prefer more detailed answers. Thus there is no single "human preference", but rather a collection of preference decisions that depend on the annotation guidelines, cultural context, and human idiosyncrasies (Conitzer et al., 2024;

Kirk et al., 2024).

Note that following Peyrard et al. (2021), as well as Chatbot Arena (Chiang et al., 2024), we generally regard the ground-truth quality of a system in terms of the Bradley-Terry model; simply put, a better system is a system that "wins" more often. Thus, in this work we do not directly consider the quality difference in system responses per instance, i.e., beyond counting wins/losses. Still, some of the aggregation methods we use (e.g., mean) implicitly reflect other perspectives on system quality.

All of our analyses are performed on heterogeneous datasets of user instructions to LLMs. Thus, while we study judges through the lens of general-purpose LLM usage, we cannot draw conclusions on judge behavior that is task-specific (or in specialized domains), nor on performance in languages other than English (Gureja et al., 2024). The issue of task, domain, and language-specific judge behavior is thus an important avenue for future work.

References

Shir Ashury Tahan, Ariel Gera, Benjamin Sznajder, Leshem Choshen, Liat Ein-Dor, and Eyal Shnarch. 2024. Label-efficient model selection for text generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8384–8402, Bangkok, Thailand. Association for Computational Linguistics.

Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, et al. 2024. LLMs instead of human judges? a large scale empirical study across 20 NLP evaluation tasks. *arXiv:2406.18403*.

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, et al. 2024. InternLM2 technical report. *arXiv:2403.17297*.

Lichang Chen, Chen Zhu, Jiuhai Chen, Davit Soselia, Tianyi Zhou, Tom Goldstein, Heng Huang, Mohammad Shoeybi, and Bryan Catanzaro. 2024. ODIN: Disentangled reward mitigates hacking in RLHF. In *Forty-first International Conference on Machine Learning*.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E Gonzalez, et al. 2024. Chatbot Arena: An open platform for evaluating LLMs by human preference. In *Forty-first International Conference on Machine Learning*.

Vincent Conitzer, Rachel Freedman, Jobst Heitzig, Wesley H Holliday, Bob M Jacobs, Nathan Lambert, Milan Mossé, Eric Pacuit, Stuart Russell, Hailey Schoelkopf, et al. 2024. Social choice should guide ai alignment in dealing with diverse human feedback. *arXiv:2404.10271*.

Daniel Deutsch, Rotem Dror, and Dan Roth. 2022. Re-examining system-level correlations of automatic summarization evaluation metrics. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 6038–6052, Seattle, United States. Association for Computational Linguistics.

Florian E Dorner, Vivian Y Nastl, and Moritz Hardt. 2024. Limits to scalable evaluation at the frontier: LLM as judge won't beat twice the data. *arXiv:2410.13341*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The Llama 3 herd of models. *arXiv:2407.21783*.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled AlpacaEval: A simple way to debias automatic evaluators. *arXiv:2404.04475*.

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

Srishti Gureja, Lester James Validad Miranda, Shayekh Bin Islam, Rishabh Maheshwary, Drishti Sharma, Gusti Winata, Nathan Lambert, Sebastian Ruder, Sara Hooker, and Marzieh Fadaee. 2024. M-RewardBench: Evaluating reward models in multilingual settings. *arXiv:2410.15522*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv:2401.04088*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Hannah Rose Kirk, Alexander Whitefield, Paul Röttger, Andrew Bean, Katerina Margatina, Juan Ciro, Rafael Mosquera, Max Bartolo, Adina Williams, He He, Bertie Vidgen, and Scott A Hale. 2024. The PRISM alignment project: What participatory, representative and individualised human feedback reveals about the subjective and multicultural alignment of large language models. *arXiv:2404.16019*.

Meelis Kull, Telmo Silva Filho, and Peter Flach. 2017. Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 623–631. PMLR.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. 2024. RewardBench: Evaluating reward models for language modeling. *arXiv:2403.13787*.

Dongryeol Lee, Yerin Hwang, Yongil Kim, Joonsuk Park, and Kyomin Jung. 2024a. Are LLM-judges robust to expressions of uncertainty? investigating the effect of epistemic markers on LLM-based evaluation. *arXiv:2410.20774*.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2024b. RLAIF vs. RLHF: Scaling reinforcement learning from human feedback with ai feedback. *arXiv:2309.00267*.

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

Rensis Likert. 1932. A technique for the measurement of attitudes. *Archives of Psychology*.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. 2024a. Skywork-reward: Bag of tricks for reward modeling in LLMs. *arXiv:2410.18451*.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. 2024b. RM-bench: Benchmarking reward models of language models with subtlety and style. *arXiv:2410.16184*.

Xingzhou Lou, Dong Yan, Wei Shen, Yuzi Yan, Jian Xie, and Junge Zhang. 2024. Uncertainty-aware reward model: Teaching reward models to know what is unknown. *arXiv:2410.00847*.

Annie Louis and Ani Nenkova. 2013. Automatically assessing machine summary content without a gold standard. *Computational Linguistics*, 39(2):267–300.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? a call for multi-prompt LLM evaluation. *arXiv:2401.00595*.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. 2024. Llm evaluators recognize and favor their own generations. In *Advances in Neural Information Processing Systems*, volume 37, pages 68772–68802. Curran Associates, Inc.

Junsoo Park, Seungyeon Jwa, Ren Meiying, Daeyoung Kim, and Sanghyuk Choi. 2024. OffsetBias: Leveraging debiased data for tuning evaluators. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1043–1067, Miami, Florida, USA. Association for Computational Linguistics.

Yotam Perlitz, Ariel Gera, Ofir Arviv, Asaf Yehudai, Elron Bandel, Eyal Shnarch, Michal Shmueli-Scheuer, and Leshem Choshen. 2024. Do these LLM benchmarks agree? Fixing benchmark evaluation with BenchBench. *arXiv:2407.13696*.

Maxime Peyrard, Wei Zhao, Steffen Eger, and Robert West. 2021. Better than average: Paired evaluation of NLP systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2301–2315, Online. Association for Computational Linguistics.

Ehud Reiter and Anja Belz. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics*, 35(4):529–558.

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

Telmo Silva Filho, Hao Song, Miquel Perello-Nieto, Raul Santos-Rodriguez, Meelis Kull, and Peter Flach. 2023. Classifier calibration: a survey on how to assess and improve predicted class probabilities. *Machine Learning*, 112(9):3211–3260.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. 2024. JudgeBench: A benchmark for evaluating LLM-based judges. *arXiv:2410.12784*.

Aman Singh Thakur, Kartik Choudhary, Venkat Srinik Ramayapally, Sankaran Vaidyanathan, and Dieuwke Hupkes. 2024. Judging the judges: Evaluating alignment and vulnerabilities in LLMs-as-judges. *arXiv:2406.12624*.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442, Singapore. Association for Computational Linguistics.

John W Tukey. 1949. Comparing individual means in the analysis of variance. *Biometrics*, pages 99–114.

Pius von Däniken, Jan Deriu, and Mark Cieliebak. 2024. A measure of the system dependence of automated metrics. *arXiv:2412.03152*.

Pius Von Däniken, Jan Deriu, Don Tuggener, and Mark Cieliebak. 2024. Favi-Score: A measure for favoritism in automated preference ratings for generative AI evaluation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4437–4454, Bangkok, Thailand. Association for Computational Linguistics.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10582–10592, Miami, Florida, USA. Association for Computational Linguistics.

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

Hui Wei, Shenghua He, Tian Xia, Andy Wong, Jingyang Lin, and Mei Han. 2024. Systematic evaluation of LLM-as-a-judge in LLM alignment tasks: Explainable metrics and diverse prompt templates. *arXiv:2408.13006*.

Wenda Xu, Guanglei Zhu, Xuandong Zhao, Liangming Pan, Lei Li, and William Wang. 2024. 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)*, pages 15474–15492, Bangkok, Thailand. Association for Computational Linguistics.

Rui Yang, Ruomeng Ding, Yong Lin, Huan Zhang, and Tong Zhang. 2024. Regularizing hidden states enables learning generalizable reward model for LLMs. In *Advances in Neural Information Processing Systems*.

Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, et al. 2024. Justice or prejudice? quantifying biases in LLM-as-a-judge. *arXiv:2410.02736*.

Asaf Yehudai, Boaz Carmeli, Yosi Mass, Ofir Arviv, Nathaniel Mills, Eyal Shnarch, and Leshem Choshen. 2024. Achieving human parity in content-grounded datasets generation. In *The Twelfth International Conference on Learning Representations*.

Lifan Yuan, Ganqu Cui, Hanbin Wang, Ning Ding, Xingyao Wang, Jia Deng, Boji Shan, Huimin Chen, Ruobing Xie, Yankai Lin, Zhenghao Liu, Bowen Zhou, Hao Peng, Zhiyuan Liu, and Maosong Sun. 2024. Advancing LLM reasoning generalists with preference trees. *arXiv:2404.02078*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc.

A Judge Score Distributions

Figure 12 depicts the score distributions, D^p, of the judges in our data.

Reward model distributions The reward models exhibit continuous score distributions. As seen in Figure 12, these distributions vary in the range of scores, as well as in the shape of the distribution. Some reward model judges have a narrow range of scores, e.g., -0.1 to 0.4, whereas in others it is much wider, e.g., -3000 to 5000. Similarly, some distributions are more symmetric while others have peaks at more extreme values. However, all distributions are uni-modal, with a single peak. Moreover, we note that the continuous nature of these judgment scores also entails an absence of ties between the judged responses.

LLM Numeric distributions As shown in Figure 12, even though the LLM judges are given a wide range of possible judgment scores ($[0-100]$), in practice they tend to prefer specific score values. This results in many ties when comparing responses from different systems.

LLM Likert distributions Similarly to the Numeric distributions, the Likert realizations put most of their probability mass on specific scores, which leads to an even greater inclination towards ties (as here they are limited to a smaller range of scores).

LLM TokenProbs distributions TokenProbs scores tend to be extreme, namely very close to either 0.0 or 1.0. Thus, in many cases the score gap between responses is extremely small. This can result in low judge robustness (see the error bars in Figure 4), as well as a higher sensitivity to the choice of aggregation method.

LLM Anchor distributions The distribution for Anchor judgments is mainly tied to the quality of the anchor system relative to the other systems. However, we see that it is also affected by the characteristics of the judge. For example, we see in Fig. 12 that Llama-3.1-8B exhibits indecision, rating most responses as comparable to those of the anchor. In addition, for some judges, the proportion of -1 scores (i.e., the response is slightly worse than the anchor) or 1 scores (the response is slightly better than the anchor) is unusually low.

B Aggregation Methods

Given the raw judgments of each judge, $j_p(R)$, there are multiple aggregation methods, a, that construct a *ranking* over all the target systems. Here, we calculate rankings using **Win-rate** aggregation, **BT** aggregation, **Mean** aggregation, and **Median** aggregation. In the following, we provide further details on each aggregation.

Mean & Median Aggregation These aggregation methods map a score for each system, s_l, by relying solely on the scores assigned to its responses by judge j_p. In other words, the mapping of $V_l^{p,a}$ by a depends only on the column corresponding to system s_l in $j_p(R)$. Accordingly, these aggregations can be viewed as an operation on the columns of the scores matrix $j_p(R)$. Specifically, for the **Mean** aggregation, $V_l^{p,a} = \frac{1}{K}\Sigma_{k=1}^{K} Score_{k,l}^{p}$. Similarly, **Median** aggregation is the median of the vector $j_p(R)_{*l}$.

We note that for realizations with discrete score distributions (see §A), many systems will likely share the same median score; in this case, the Median aggregation method fails to separate the systems. Hence, Table 2 contains only a handful of LLM judges with Median aggregation, all using the *TokenProbs* realization.

Win-rate Aggregation This aggregation maps each system based on its proportion of wins over other systems, averaged over all instructions $i_k \in I$. Formally, $V_b^{p,a} = \frac{1}{K}\Sigma_{k=1}^{K}\frac{1}{L-1}\Sigma_{l=1,l\neq b}^{L}\mathbb{I}(Score_{k,b}^{p} > Score_{k,l}^{p})$, where $\mathbb{I}(\cdot)$ denotes the indicator function.

Bradley-Terry Aggregation Following Chiang et al. (2024), we use the vector of Bradley-Terry (BT) coefficients (Bradley and Terry, 1952) as system scores.

For calculating the BT scores we use the implementation of the Chatbot Arena official notebook[7]. Whereas Chiang et al. (2024) apply this method for battles between responses with a human judge, we apply it over our LLM-based judge data, i.e., each "battle" is a comparison between the judge scores $Score_{k,a}^{p}$, $Score_{k,b}^{p}$ for a response generated by systems s_a and s_b.

When there are no ties, e.g., for the reward model judges, this aggregation produces similar rankings to the win-rate aggregation.

C Chatbot Arena Data

The data for the Chatbot Arena LLM leaderboard (`https://lmarena.ai`) consists of "battles" between systems over the same instructions. In these

[7] Arena official notebook

battles, users indicate a preference (or a tie) between a pair of responses generated by different LLMs (Zheng et al., 2023; Chiang et al., 2024).

We use their public data file from August 2024[8], and follow the official notebook[7] to extract the raw data, deduplicate it, and calculate the overall system rankings. This dataset includes the human preference judgments and names of the participating systems, but not the instructions or system responses for the battles.

Here we limit the analysis to the *English Hard Prompts* subset of their data[9] (300K battles). Notably, Arena Hard was specifically designed to match the distribution of user instructions in the *English Hard Prompts* subset, as described by Li et al. (2024). We follow their code to construct a full system ranking based on these 300K battles, using Bradley-Terry coefficients. This yields a score for each system in their data, including 59 systems that are also in our system responses data (§4.1)

Out of this full English Hard data, we also extract a total of 113K battles that were not judged by humans as ties, and that are between pairs of systems which appear in our responses data. We then use those to calculate win-rates between pairs of systems (§E), yielding a total of 968 system pairwise win-rates. Note that the Chatbot Arena data does not contain battles between every possible pairing of systems, and thus we do not have win-rates for all combinations of the 59 systems under consideration. In addition, we limit the analysis to system pairs with at least 10 non-tied battles.

D Statistical Analysis of Judge Performance

In §5 and Table 2 we report results of agreement with the gold ranking (τ) for various judge pipelines. Each pipeline consists of a chosen judge model, a realization (§4.2.2) and an aggregation method (§4.3, App. B).

We focus on the LLM judges and perform a three-way ANOVA (analysis of variance), with the ranking correlation τ as a dependent variable and the *model*, *realization* and *aggregation* as factors. In addition to the variance analysis estimating the effects of these factors, we perform post-hoc pairwise comparisons to ask whether certain configurations (i.e., a specific realization/aggregation) outperform the others. We conduct all analyses using IBM SPSS Statistics v30.0.

The ANOVA shows that both the judge model and the realization have a strong influence on τ, with an effect size (*Partial Eta-Squared*) of $\eta^2 = 0.81$ for the judge model ($p < 0.001$; $F = 36.0$), $\eta^2 = 0.51$ for the realization ($p < 0.001$; $F = 26.6$), and $\eta^2 = 0.78$ for the interaction effect between model and realization ($p < 0.001$; $F = 10.1$). In contrast, the aggregation methods were not found to have a significant effect on τ ($\eta^2 = 0.02$; $p > 0.5$).

We also perform Tukey's HSD (Tukey, 1949) post-hoc tests to compare the means of the variables. The analysis indicates that the both the Numeric (mean $\tau = 0.75$; $\sigma_\tau = 0.06$) and Likert ($\tau = 0.74$; $\sigma_\tau = 0.07$) realizations are significantly better than the Anchor ($\tau = 0.71$; $\sigma_\tau = 0.07$) and TokenProbs ($\tau = 0.68$; $\sigma_\tau = 0.13$) realizations (all p values $<= 0.002$). The differences between aggregation methods are not statistically significant.

E Pairwise Win-Rates

We denote the win-rate of system s_a over system s_b as $WR(s_a, s_b)^p$ where p denotes the judge upon which the win-rate was calculated, and $p \in J \cup \{g\}$, where g stands for human gold data.

The win-rate of system s_a over system s_b according to judge j_p over the set of instances I is calculated as the proportion of instances where the score given by j_p to the response generated by s_a surpasses that of system s_b, where ties are excluded. Namely $WR^p(s_a, s_b) = \frac{1}{K - |T^p_{a,b}|} \Sigma_{k=1}^{K} \mathbb{I}(Score^p_{k,a} > Score^p_{k,b})$ Where $T^p_{a,b} = \{i_k | Score^p_{k,a} = Score^p_{k,b}\}$, and $\mathbb{I}(\cdot)$ denotes the indicator function. Notice that $WR^p(s_a, s_b) = 1 - WR^p(s_b, s_a)$.

To quantify the agreement between the judge and gold win-rates we also define an *Accuracy* metric. This measures the proportion of pairs where the judge pairwise system preference decisions are in agreement with those of the human gold-data. In other words, we want to count the pairs that appear in the first and third quadrants in Figure 5; namely, the pairs where the judge and gold win-rate are both bigger than 0.5, or the pairs where both are lower than 0.5, representing agreement on the winning system. For that, we denote all the pairs of systems we have in the gold data as $\{s_{a^m}, s_{b^m}\}_{m=1}^{M}$. Now

[8]Chatbot Arena data
[9]Chatbot Arena Hard Prompts

the *Accuracy* is defined as follows:

$$Acc^{p}_{WR} = \frac{1}{M}\Sigma_{m=1}^{M}\mathbb{I}(\mathbb{I}(WR^p(s_{a^m}, s_{b^m}) > 0.5) \\ = \mathbb{I}(WR^g(s_{a^m}, s_{b^m}) > 0.5))$$

Additionally, we define a second metric, the *Mean Squared Error* over all win-rate pairs.

$$MSE^{m}_{WR} = \frac{1}{M}\Sigma_{m=1}^{M}(WR^g(s_{a^m}, s_{b^m}) \\ -WR^p(s_{a^m}, s_{b^m}))^2.$$

The Acc^{p}_{WR} scores are in high agreement with the JuStRank judge ranking quality scores τ (Pearson correlation of $r = 0.96$ for the BT aggregation, $r = 0.79$ for the Mean aggregation). This highlights the direct link between judges' ability to rank systems and their performance on pairwise system preference.

The MSE^{p}_{WR} scores have a low correlation with the JuStRank judge τ scores ($r = -0.19$ for the BT aggregation, $r = -0.07$ for the Mean aggregation). This can be explained by the decisiveness effect (§6.1), where judges deviate substantially from the gold win-rate, but mostly toward the stronger system in the pair.

F Beta Distribution Fit

Following Kull et al. (2017), we model the relation between judge and gold win-rates using the cumulative distribution function (CDF) of the Beta distribution. We parameterize the distribution such that both shape parameters α and β are equal ($\alpha = \beta$).

The CDF of the Beta distribution, defined over the interval $[0, 1]$, for $\alpha = \beta \in [0, \infty]$ provides a wide range of function fits: a linear $y = x$ fit for $\alpha = 1$, a sigmoidal fit for larger α values, and approaching a step function as $\alpha \to \infty$. These attributes make it particularly suited for our data characteristics.

Given a set of data points $\{(WR^p(s_{a^m}, s_{b^m}), WR^g(s_{a^m}, s_{b^m})\}_{m=1}^{M}$, where $WR^p(s_{a^m}, s_{b^m}) \in [0, 1]$ represents the judge win-rate and $WR^g(s_{a^m}, s_{b^m}) \in [0, 1]$ denotes the gold win-rate between system, s_{a^m} and s_{b^m}. We fit the Beta CDF by optimizing the shape parameter α. The optimization objective is minimizing the sum of absolute errors (SAE) between the judge win-rate, $WR^p(s_{a^m}, s_{b^m})$, and the predicted values from the Beta CDF. In order to capture the behavior across the entire range of win-rates, we weight the errors by the distance of WR^p from 0.5:

$$\text{SAE} = \sum_{m=1}^{M} \gamma(WR^p(s_{a^m}, s_{b^m})) \cdot \Big| WR^p(s_{a^m}, s_{b^m}) \\ - F_{\text{Beta}}(WR^g(s_{a^m}, s_{b^m}); \alpha) \Big|$$

where $F_{\text{Beta}}(x; \alpha)$ denotes the Beta CDF with shape parameters $\alpha = \beta$, and γ is the distance of WR^p from 0.5.

The optimization was performed using the `scipy.optimize.minimize`[10] function, with the parameter (α) constrained to a reasonable range $[0.1, 10000]$. This approach efficiently identified the best-fit parameter (α).

The resulting Beta CDF closely captures the empirical data distribution, as validated both quantitatively, through low SAE, and qualitatively via visual inspection. Figure 11 depicts the fitted Beta CDF curve and the observed data points, demonstrating the effectiveness of this approach for modeling the judges' predicted win-rate distribution.

[10] SciPy Documentation for scipy.optimize.minimize

G LLM Judge Prompts

Below we list the prompts we use for each LLM judge realization (§4.2.2).

Numeric

Here is a user input and a model response. On a scale of 0 to 100, to what extent is this a good response for the given input? Reply with your rating score without any preceding explanation. Input: *[user instruction]*
Response: *[system response]*
Rating (0-100):

Likert

Here is a user input and a model response. To what extent is this a good response for the given input? Provide a rating from one of the following choices: 'Very Bad', 'Bad', 'Mediocre', 'Good', 'Very Good'. Reply using the format of [[rating]], for example: '[[Mediocre]]'
Input: *[user instruction]*
Response: *[system response]*
Rating:

TokenProbs

Here is a user input and a model response. Is this a good response for the given input? Answer with only yes/no. Input: *[user instruction]*
Response: *[system response]*
Good response? (Yes/No):

Anchor

Here is a user input and responses from two assistants, A and B. Which response is better? You must output only one of the following choices as your final verdict with a label:

1. Assistant A is significantly better: [[A>>B]]
2. Assistant A is slightly better: [[A>B]]
3. Tie, relatively the same: [[A=B]]
4. Assistant B is slightly better: [[B>A]]
5. Assistant B is significantly better: [[B>>A]

Example output: "My final verdict is tie: [[A=B]]".

<|User Prompt|>
[user instruction]

<|The Start of Assistant A's Answer|>
[system response]
<|The End of Assistant A's Answer|>

<|The Start of Assistant B's Answer|>
[anchor system response]
<|The End of Assistant B's Answer|>
Final Verdict:

Judge Model	Realization	Aggregation	Agreement (τ) w/ Gold Ranking
Qwen2.5-72B-Instruct	Likert	Win-Rate	.827
URM-LLaMa-3.1-8B	Reward	Mean	.823
GPT-4o-2024-11-20	Anchor	Mean	.822
URM-LLaMa-3.1-8B	Reward	BT	.819
Qwen2.5-72B-Instruct	Likert	BT	.817
URM-LLaMa-3.1-8B	Reward	Win-Rate	.816
Qwen2.5-72B-Instruct	Numeric	BT	.814
GPT-4o-2024-11-20	Anchor	Win-Rate	.814
Qwen2.5-72B-Instruct	Numeric	Win-Rate	.813
Llama-3-1-405b-instruct-fp8	Numeric	Mean	.812
Llama-3-1-405b-instruct-fp8	Numeric	Win-Rate	.812
Mistral-large-instruct-2407	Likert	BT	.811
GPT-4o-2024-11-20	Anchor	BT	.809
Mistral-large-instruct-2407	Numeric	BT	.809
URM-LLaMa-3.1-8B	Reward	Median	.809
GPT-4o-mini-2024-07-18	Numeric	Win-Rate	.807
Llama-3-1-405b-instruct-fp8	Numeric	BT	.805
GPT-4o-mini-2024-07-18	Numeric	BT	.804
Mistral-large-instruct-2407	Numeric	Win-Rate	.802
Qwen2.5-72B-Instruct	Likert	Mean	.801
ArmoRM-Llama3-8B-v0.1	Reward	Mean	.800
Qwen2.5-72B-Instruct	Anchor	Mean	.799
GPT-4o-mini-2024-07-18	Likert	BT	.798
Llama-3-1-70b-instruct	Numeric	Win-Rate	.798
Llama-3-1-70b-instruct	Numeric	BT	.798
Mistral-large-instruct-2407	Likert	Win-Rate	.798
Qwen2.5-72B-Instruct	Anchor	BT	.794
Llama-3-1-405b-instruct-fp8	Likert	Win-Rate	.793
Llama-3-1-70b-instruct	TokenProbs	Win-Rate	.793
GPT-4o-mini-2024-07-18	Likert	Win-Rate	.793
ArmoRM-Llama3-8B-v0.1	Reward	Median	.793
Llama-3-1-405b-instruct-fp8	Likert	BT	.787
Mistral-large-instruct-2407	Anchor	Win-Rate	.786
Skywork-Llama-3.1-8B-v0.2	Reward	Mean	.786
Qwen2.5-72B-Instruct	Anchor	Win-Rate	.786
Mistral-large-instruct-2407	Likert	Mean	.782
GPT-4o-mini-2024-07-18	Numeric	Mean	.781
Skywork-Llama-3.1-8B-v0.2	Reward	Win-Rate	.780
Llama-3-1-405b-instruct-fp8	Likert	Mean	.780
Skywork-Llama-3.1-8B-v0.2	Reward	BT	.778
Llama-3.1-8B-Instruct	TokenProbs	Mean	.778
Qwen2.5-72B-Instruct	TokenProbs	BT	.777
Llama-3.1-8B-Instruct	TokenProbs	Median	.776
Mixtral-8x22B-instruct-v0.1	Numeric	BT	.776
Llama-3-1-70b-instruct	TokenProbs	Median	.776
GPT-4o-2024-11-20	Numeric	BT	.774
GPT-4o-mini-2024-07-18	Likert	Mean	.773
Qwen2.5-72B-Instruct	Numeric	Mean	.773

GPT-4o-2024-11-20	Likert	BT	.773
GPT-4o-2024-11-20	Numeric	Win-Rate	.771
Llama-3-OffsetBias-RM-8B	Reward	Win-Rate	.765
Llama-3-1-70b-instruct	TokenProbs	BT	.765
Llama-3-OffsetBias-RM-8B	Reward	BT	.765
Skywork-Llama-3.1-8B-v0.2	Reward	Median	.764
Llama-3-1-70b-instruct	TokenProbs	Mean	.764
Mistral-large-instruct-2407	Anchor	Mean	.764
Llama-3-1-70b-instruct	Numeric	Mean	.764
ArmoRM-Llama3-8B-v0.1	Reward	BT	.763
ArmoRM-Llama3-8B-v0.1	Reward	Win-Rate	.762
Llama-3-OffsetBias-RM-8B	Reward	Median	.759
GPT-4o-mini-2024-07-18	TokenProbs	Win-Rate	.759
GPT-4o-2024-11-20	Likert	Win-Rate	.758
Llama-3-OffsetBias-RM-8B	Reward	Mean	.757
Mixtral-8x22B-instruct-v0.1	Numeric	Win-Rate	.756
GPT-4o-mini-2024-07-18	TokenProbs	BT	.752
Qwen2.5-72B-Instruct	TokenProbs	Median	.752
Mistral-large-instruct-2407	Numeric	Mean	.750
Llama-3-70b-instruct	Numeric	BT	.749
Qwen2.5-72B-Instruct	TokenProbs	Win-Rate	.748
Llama-3-1-405b-instruct-fp8	Anchor	Win-Rate	.748
Llama-3-1-70b-instruct	Likert	Mean	.746
GPT-4o-2024-11-20	Likert	Mean	.744
Llama-3.1-8B-Instruct	TokenProbs	Win-Rate	.744
Llama-3-1-405b-instruct-fp8	Anchor	Mean	.744
Llama-3.1-8B-Instruct	TokenProbs	BT	.741
Llama-3-1-405b-instruct-fp8	TokenProbs	Win-Rate	.741
GPT-4o-mini-2024-07-18	TokenProbs	Mean	.741
Mixtral-8x22B-instruct-v0.1	Likert	BT	.738
GPT-4o-2024-11-20	Numeric	Mean	.738
Llama-3-1-405b-instruct-fp8	TokenProbs	Median	.737
Llama-3.1-8B-Instruct	Likert	Mean	.736
Llama-3-70b-instruct	Numeric	Win-Rate	.733
Llama-3-1-405b-instruct-fp8	TokenProbs	Mean	.733
Llama-3-1-70b-instruct	Likert	Win-Rate	.732
Mixtral-8x22B-instruct-v0.1	Likert	Win-Rate	.732
Qwen2.5-72B-Instruct	TokenProbs	Mean	.732
Internlm2-7b-reward	Reward	Mean	.731
Llama-3-1-405b-instruct-fp8	Anchor	BT	.730
Mistral-large-instruct-2407	TokenProbs	Mean	.730
Internlm2-20b-reward	Reward	Mean	.728
Mistral-large-instruct-2407	Anchor	BT	.725
Internlm2-20b-reward	Reward	Median	.724
GPT-4o-mini-2024-07-18	TokenProbs	Median	.723
Llama-3.1-8B-Instruct	Likert	BT	.723
Llama-3-1-70b-instruct	Likert	BT	.722
Internlm2-7b-reward	Reward	Median	.721
Mixtral-8x22B-instruct-v0.1	Likert	Mean	.719
Internlm2-7b-reward	Reward	Win-Rate	.717
Internlm2-20b-reward	Reward	BT	.717
Mixtral-8x22B-instruct-v0.1	TokenProbs	Win-Rate	.717

Llama-3-1-70b-instruct	Anchor	Win-Rate	.716
GRM-Llama3.2-3B	Reward	Mean	.716
Internlm2-20b-reward	Reward	Win-Rate	.716
Mixtral-8x22B-instruct-v0.1	Numeric	Mean	.715
Llama-3-1-70b-instruct	Anchor	Mean	.714
GRM-Llama3.2-3B	Reward	Win-Rate	.712
Internlm2-7b-reward	Reward	BT	.712
GRM-Llama3.2-3B	Reward	BT	.711
GRM-Llama3.2-3B	Reward	Median	.706
GPT-4o-2024-11-20	TokenProbs	Median	.704
Llama-3-70b-instruct	Numeric	Mean	.704
Mixtral-8x22B-instruct-v0.1	TokenProbs	BT	.702
GPT-4o-2024-11-20	TokenProbs	Mean	.701
GPT-4o-2024-11-20	TokenProbs	BT	.700
Llama-3-70b-instruct	Likert	BT	.698
Llama-3-70b-instruct	TokenProbs	Win-Rate	.696
GPT-4o-2024-11-20	TokenProbs	Win-Rate	.696
Llama-3.1-8B-Instruct	Anchor	Mean	.695
Llama-3.1-8B-Instruct	Likert	Win-Rate	.694
Llama-3-1-70b-instruct	Anchor	BT	.688
Llama-3-70b-instruct	Likert	Win-Rate	.681
Llama-3.1-8B-Instruct	Numeric	Mean	.680
Llama-3-70b-instruct	Likert	Mean	.678
Llama-3.1-8B-Instruct	Anchor	BT	.677
GPT-4o-mini-2024-07-18	Anchor	Mean	.675
Llama-3-1-405b-instruct-fp8	TokenProbs	BT	.672
Llama-3.1-8B-Instruct	Numeric	BT	.668
GPT-4o-mini-2024-07-18	Anchor	Win-Rate	.668
Llama-3-70b-instruct	Anchor	Mean	.667
Llama-3-70b-instruct	TokenProbs	Mean	.666
Mixtral-8x22B-instruct-v0.1	Anchor	Mean	.665
Llama-3-70b-instruct	TokenProbs	BT	.663
GPT-4o-mini-2024-07-18	Anchor	BT	.659
Mixtral-8x7B-instruct-v0.1	Numeric	BT	.656
Mixtral-8x7B-instruct-v0.1	Anchor	BT	.655
Mixtral-8x22B-instruct-v0.1	TokenProbs	Mean	.650
Eurus-RM-7b	Reward	Median	.643
Eurus-RM-7b	Reward	Mean	.641
Mixtral-8x22B-instruct-v0.1	Anchor	BT	.641
Llama-3.1-8B-Instruct	Anchor	Win-Rate	.639
Llama-3-70b-instruct	Anchor	Win-Rate	.638
Llama-3-70b-instruct	Anchor	BT	.633
Llama-3.1-8B-Instruct	Numeric	Win-Rate	.632
Eurus-RM-7b	Reward	Win-Rate	.629
Eurus-RM-7b	Reward	BT	.628
Mixtral-8x7B-instruct-v0.1	Numeric	Win-Rate	.626
Mixtral-8x7B-instruct-v0.1	Numeric	Mean	.626
Mixtral-8x7B-instruct-v0.1	Anchor	Win-Rate	.622
Mixtral-8x22B-instruct-v0.1	Anchor	Win-Rate	.612
Mixtral-8x7B-instruct-v0.1	Anchor	Mean	.610
Mixtral-8x7B-instruct-v0.1	Likert	BT	.590
Mixtral-8x7B-instruct-v0.1	Likert	Mean	.585

Mixtral-8x7B-instruct-v0.1	Likert	Win-Rate	.543
Mixtral-8x7B-instruct-v0.1	TokenProbs	BT	.427
Mistral-large-instruct-2407	TokenProbs	Win-Rate	.417
Mixtral-8x7B-instruct-v0.1	TokenProbs	Mean	.411
Mixtral-8x7B-instruct-v0.1	TokenProbs	Win-Rate	.371
Mistral-large-instruct-2407	TokenProbs	BT	.369
Mistral-large-instruct-2407	TokenProbs	Median	.363

Table 2: **Judges by ranking performance**. The judges are sorted by the Kendall's Tau correlation between their overall system ranking and the gold ranking from Chatbot Arena (§4.4).

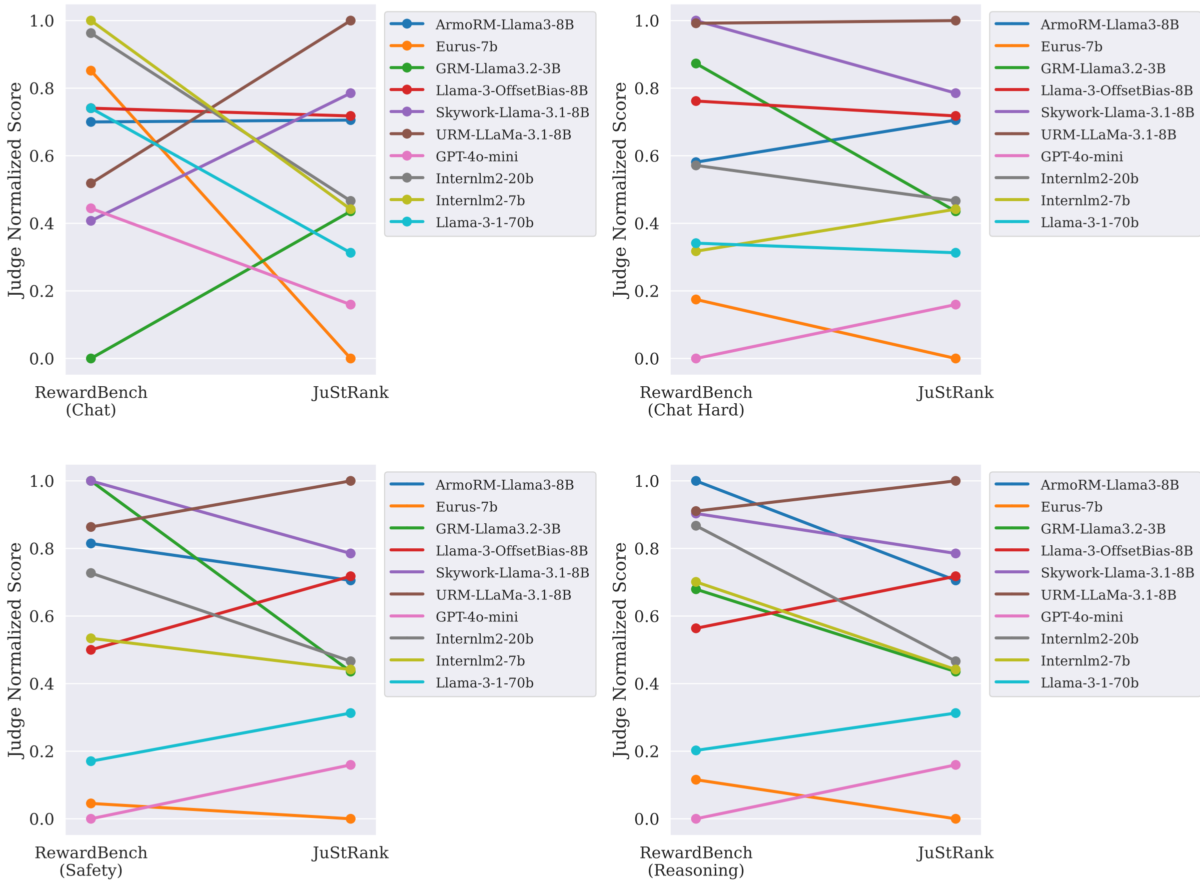


Figure 8: **Comparison to RewardBench**. The plot depicts the relative performance of judges present in both JuStRank and RewardBench (Lambert et al., 2024). For comparison, we perform Min-Max normalization over the judge performance scores (*accuracy* for RewardBench, *Kendall's Tau* for our results). The results shown are for the BT aggregation method; the LLM judges use the *Anchor* realization, which is closest to the setting in RewardBench. Each panel portrays a different subset of RewardBench.

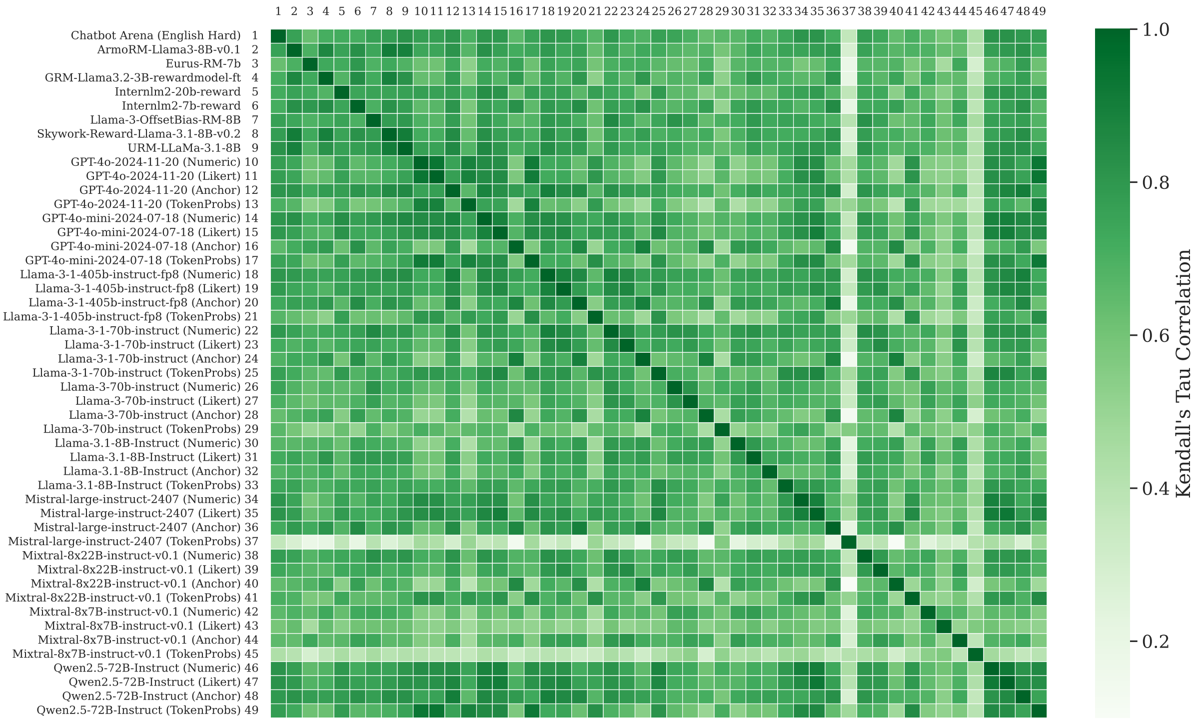
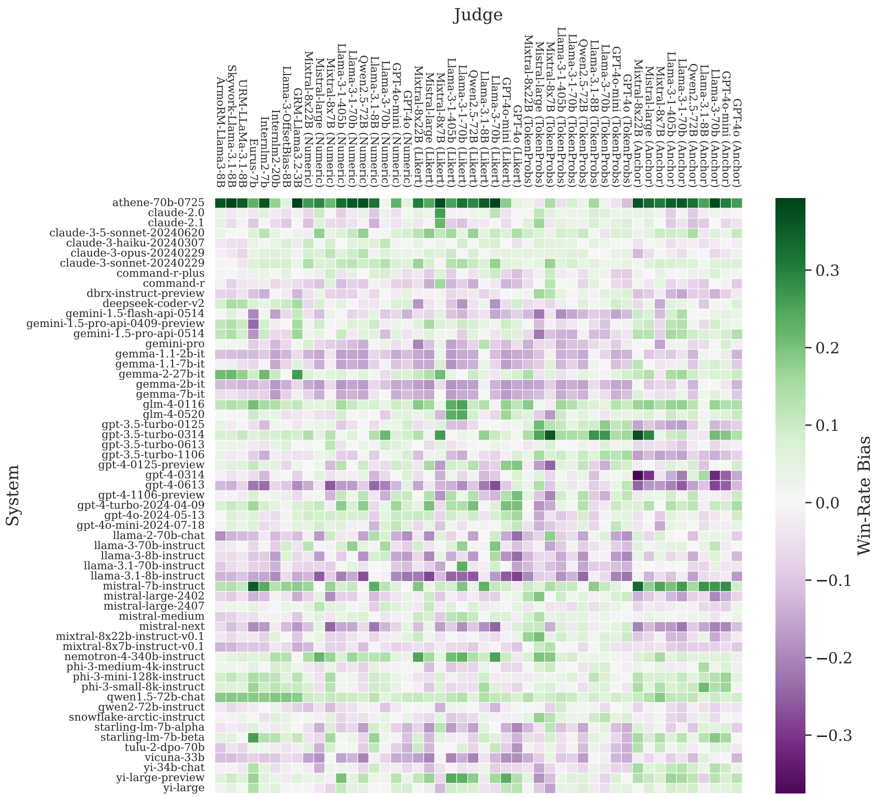


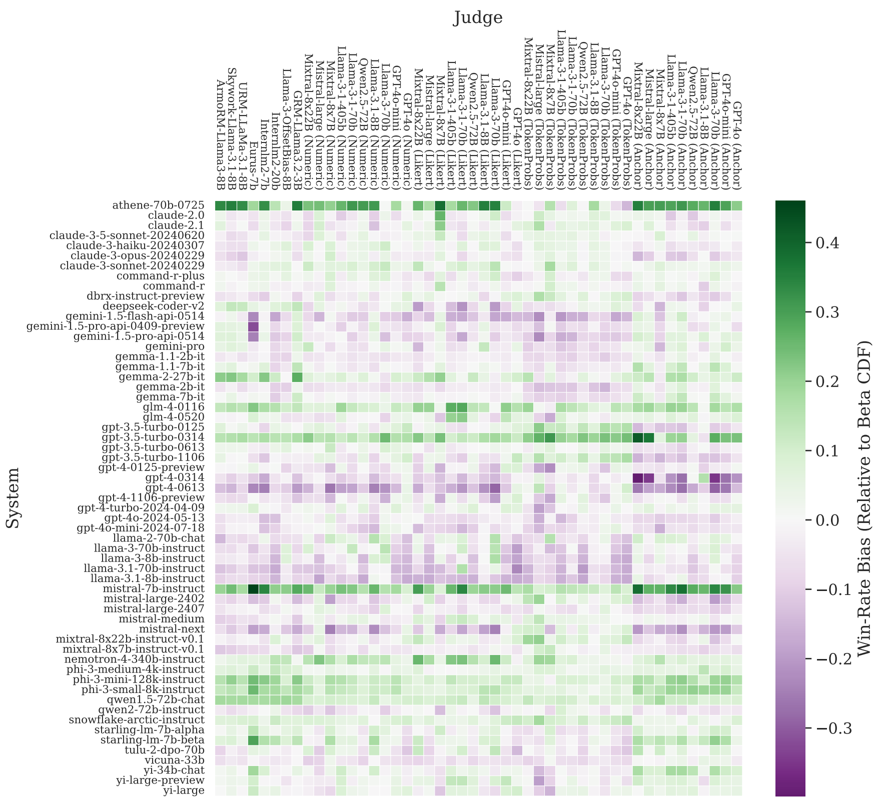
Figure 9: **Judge Correlations**. Kendall's Tau correlations between the system rankings produced by the different judge realizations, using the BT aggregation method. The first row/column denotes correlations with the reference ranking from Chatbot Arena.

Judge	Self-bias	Significance p-value
GPT-4o-mini-2024-07-18 (Anchor)	-0.05	–
GPT-4o-mini-2024-07-18 (Likert)	-0.04	–
GPT-4o-mini-2024-07-18 (Numeric)	$+0.03$	> 0.5 (N.S.)
GPT-4o-mini-2024-07-18 (TokenProbs)	$+0.06$	0.13 (N.S.)
Llama-3-1-70b-instruct (Anchor)	-0.05	–
Llama-3-1-70b-instruct (Likert)	$+0.16$	$7.1e-03$
Llama-3-1-70b-instruct (Numeric)	-0.00	–
Llama-3-1-70b-instruct (TokenProbs)	-0.03	–
Llama-3-70b-instruct (Anchor)	$+0.09$	$4.7e-04$
Llama-3-70b-instruct (Likert)	$+0.15$	$8.4e-08$
Llama-3-70b-instruct (Numeric)	$+0.14$	$1.8e-13$
Llama-3-70b-instruct (TokenProbs)	-0.01	–
Llama-3.1-8B-Instruct (Anchor)	-0.07	–
Llama-3.1-8B-Instruct (Likert)	-0.04	–
Llama-3.1-8B-Instruct (Numeric)	$+0.02$	> 0.5 (N.S.)
Llama-3.1-8B-Instruct (TokenProbs)	-0.04	–
Mistral-large-instruct-2407 (Anchor)	-0.07	–
Mistral-large-instruct-2407 (Likert)	$+0.02$	> 0.5 (N.S.)
Mistral-large-instruct-2407 (Numeric)	$+0.06$	0.33 (N.S.)
Mistral-large-instruct-2407 (TokenProbs)	$+0.01$	> 0.5 (N.S.)

Table 3: **Judge self-bias.** The table shows the self-bias values for LLM judge realizations, i.e., the value of the corrected bias ${B'}_{s_a}^{p}$ (§6.2) where the LLM judge p and system s_a correspond to the same underlying LLM. For positive self-bias values we test the statistical significance using paired t-tests (one-sided, with Bonferroni correction). N.S.: non-significant ($p > 0.05$).



(a)



(b)

Figure 10: **System-specific judge biases**. The heat maps depict the win-rate biases of various judges towards specific systems (§6.2), with respect to the ground-truth win-rates from Chatbot Arena. (a): Bias w.r.t. the raw ground-truth win-rates WR^g; (b): Bias w.r.t. the fit value for the gold win-rate $WR^{g'}$ on the beta distribution fit (App. F) for each judge.

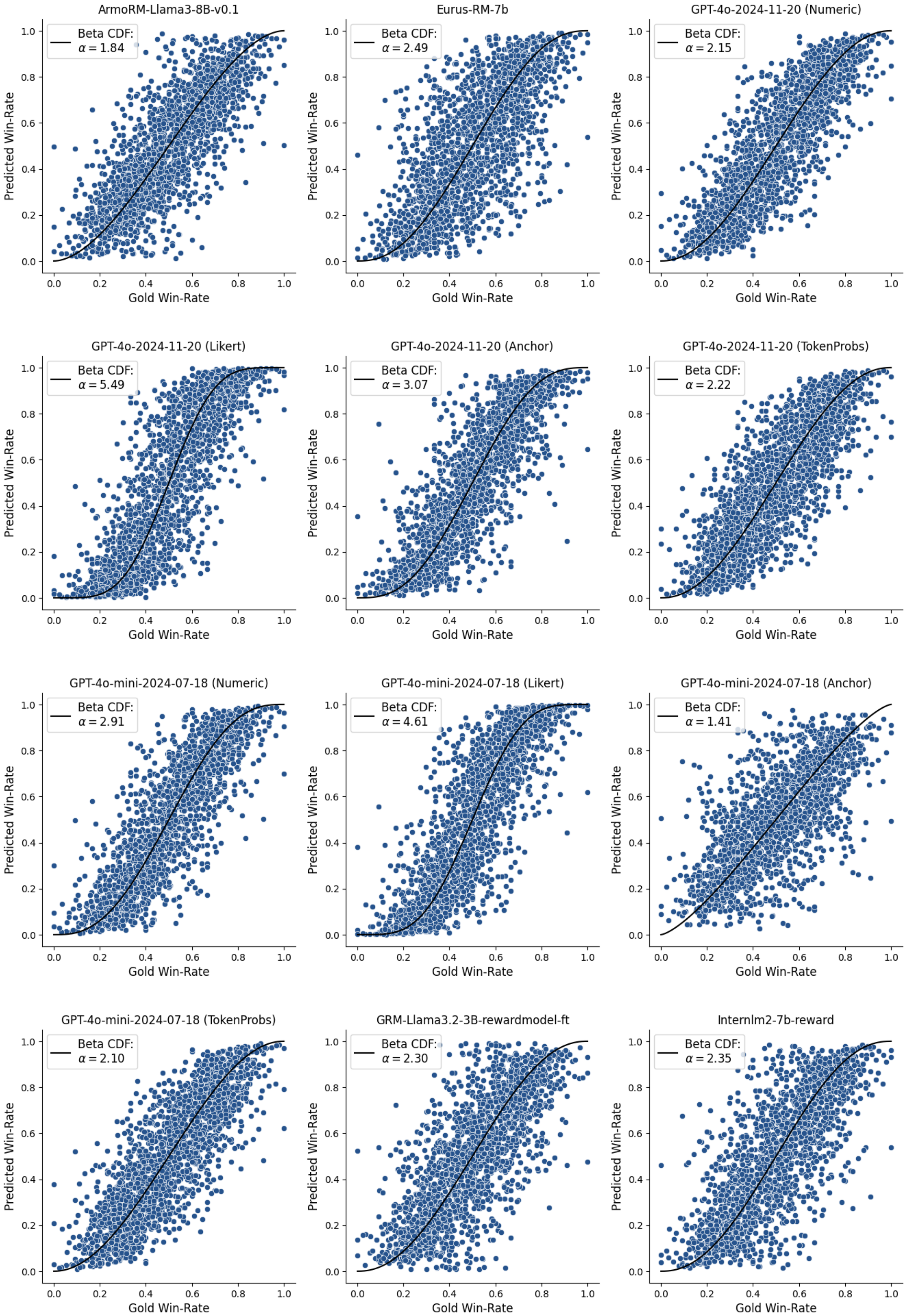


Figure 11: **Beta distribution fit of pairwise win-rates** *(Part 1/4)*

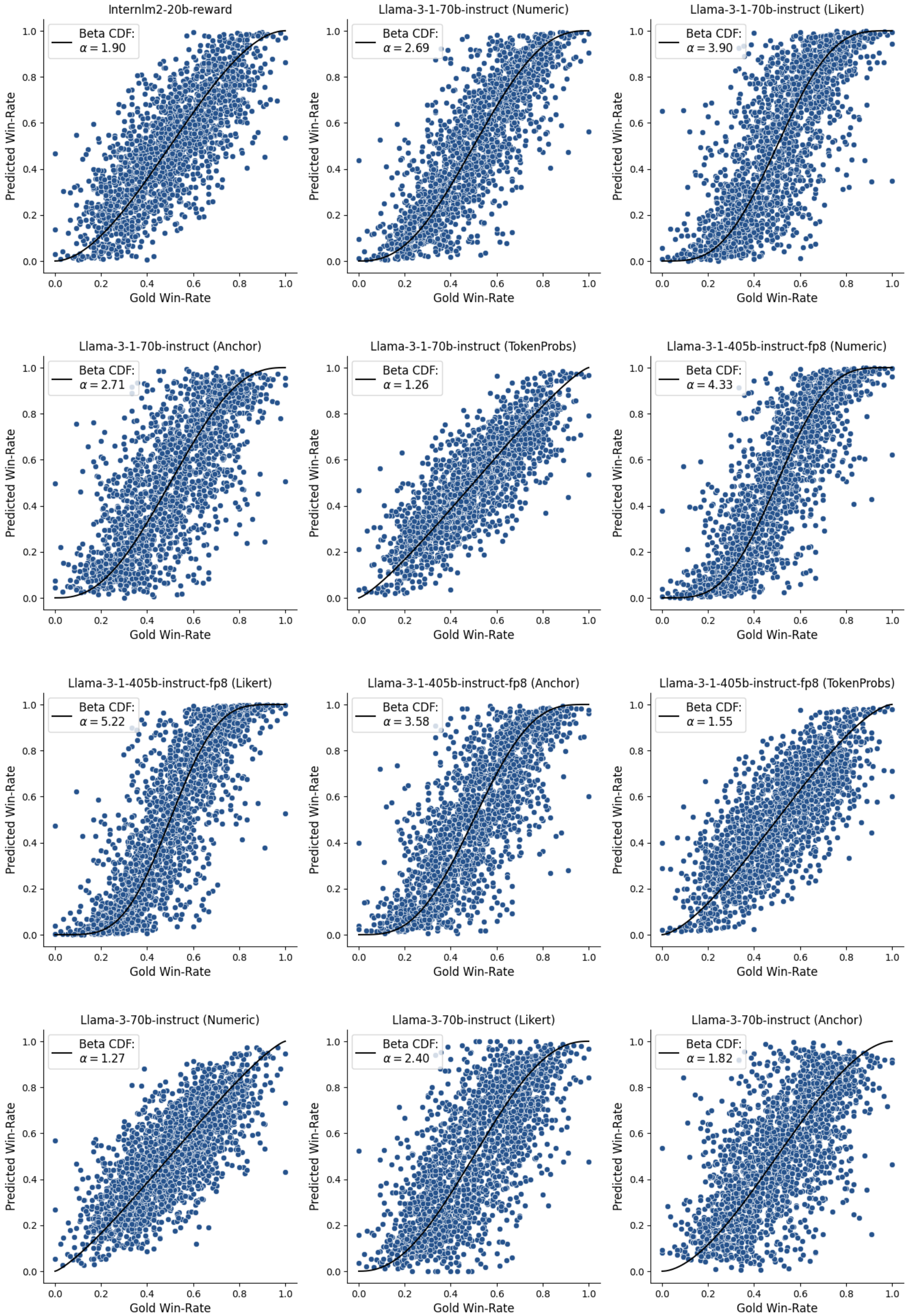


Figure 11: **Beta distribution fit of pairwise win-rates** *(Part 2/4)*

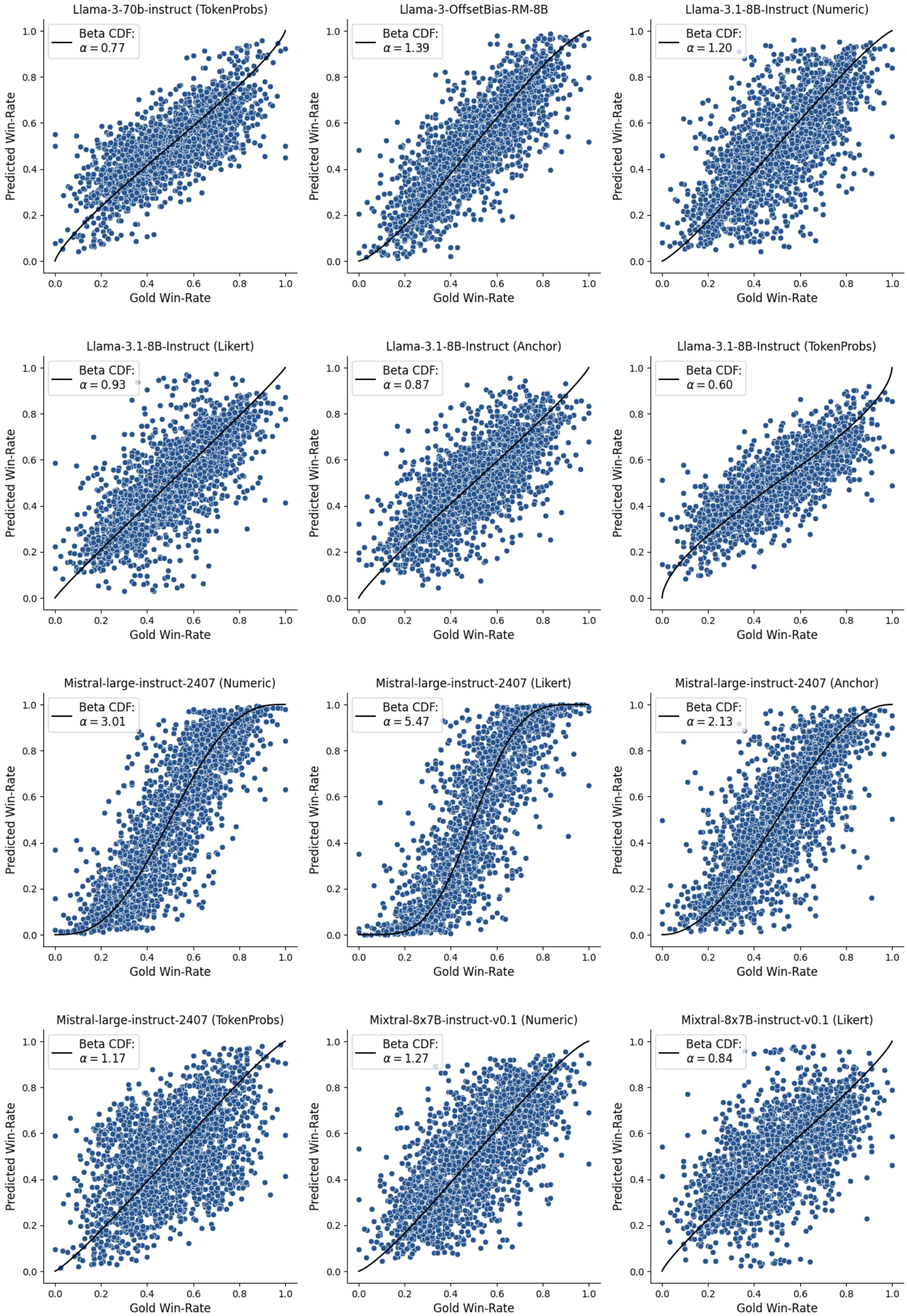


Figure 11: **Beta distribution fit of pairwise win-rates** *(Part 3/4)*

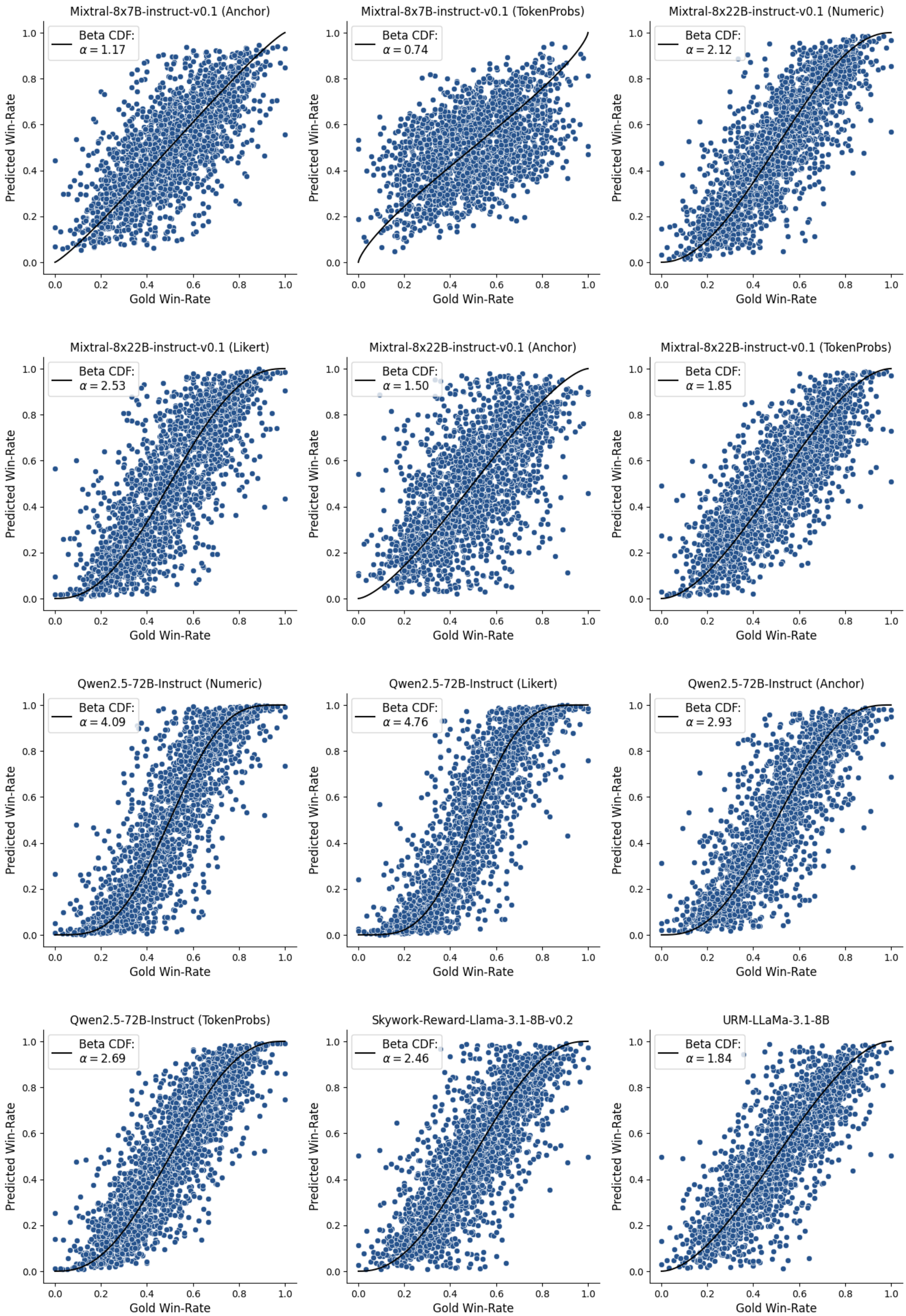


Figure 11: **Beta distribution fit of pairwise win-rates** *(Part 4/4)*. Each point represents the win-rate between a pair of systems, $WR(s_a, s_b)$; the curve and α value describe a fit to the beta probability distribution. Refer to Appendix F for details.

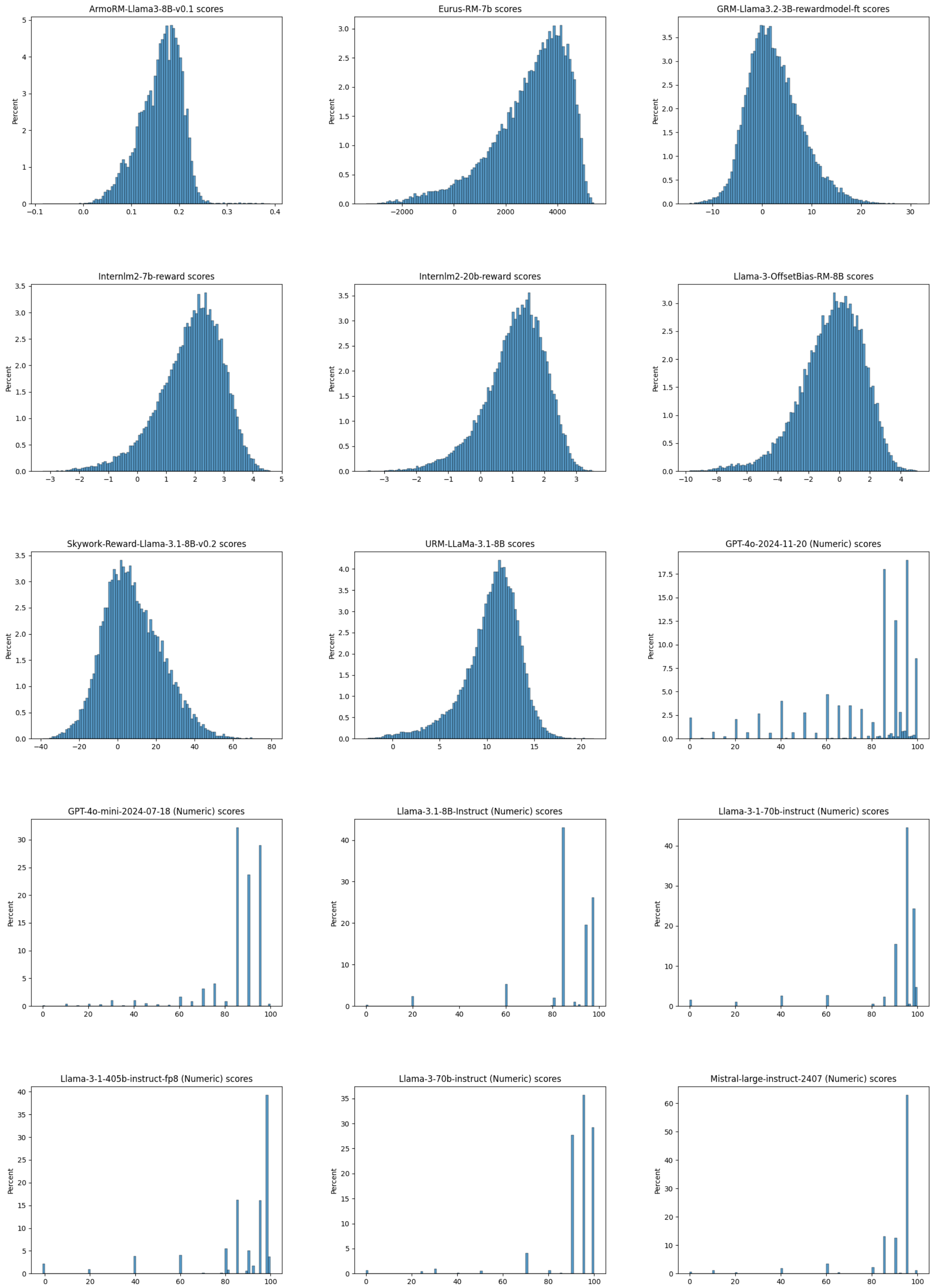


Figure 12: **Judge score distributions** *(Part 1/3)*

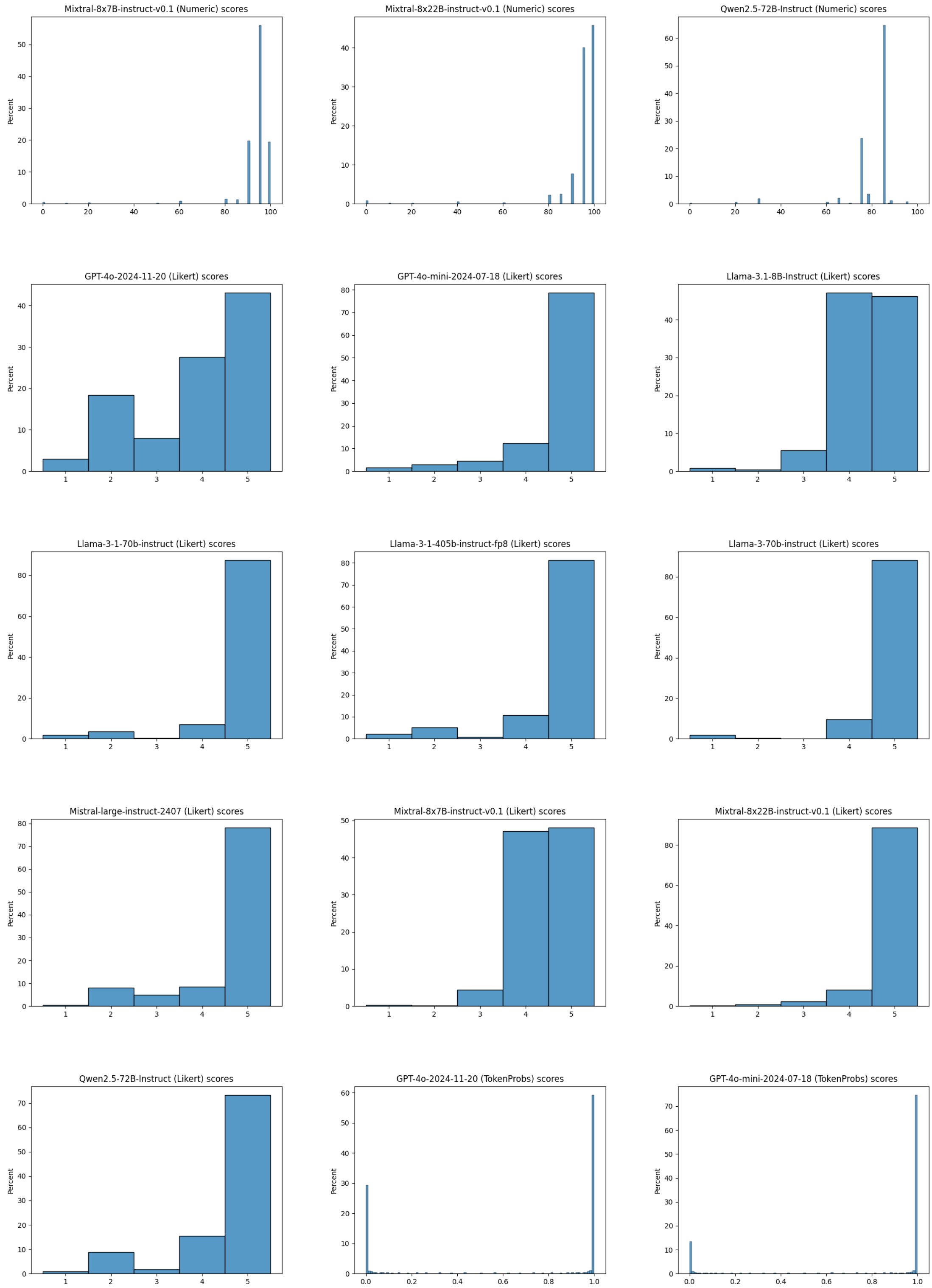


Figure 12: **Judge score distributions** *(Part 2/3)*

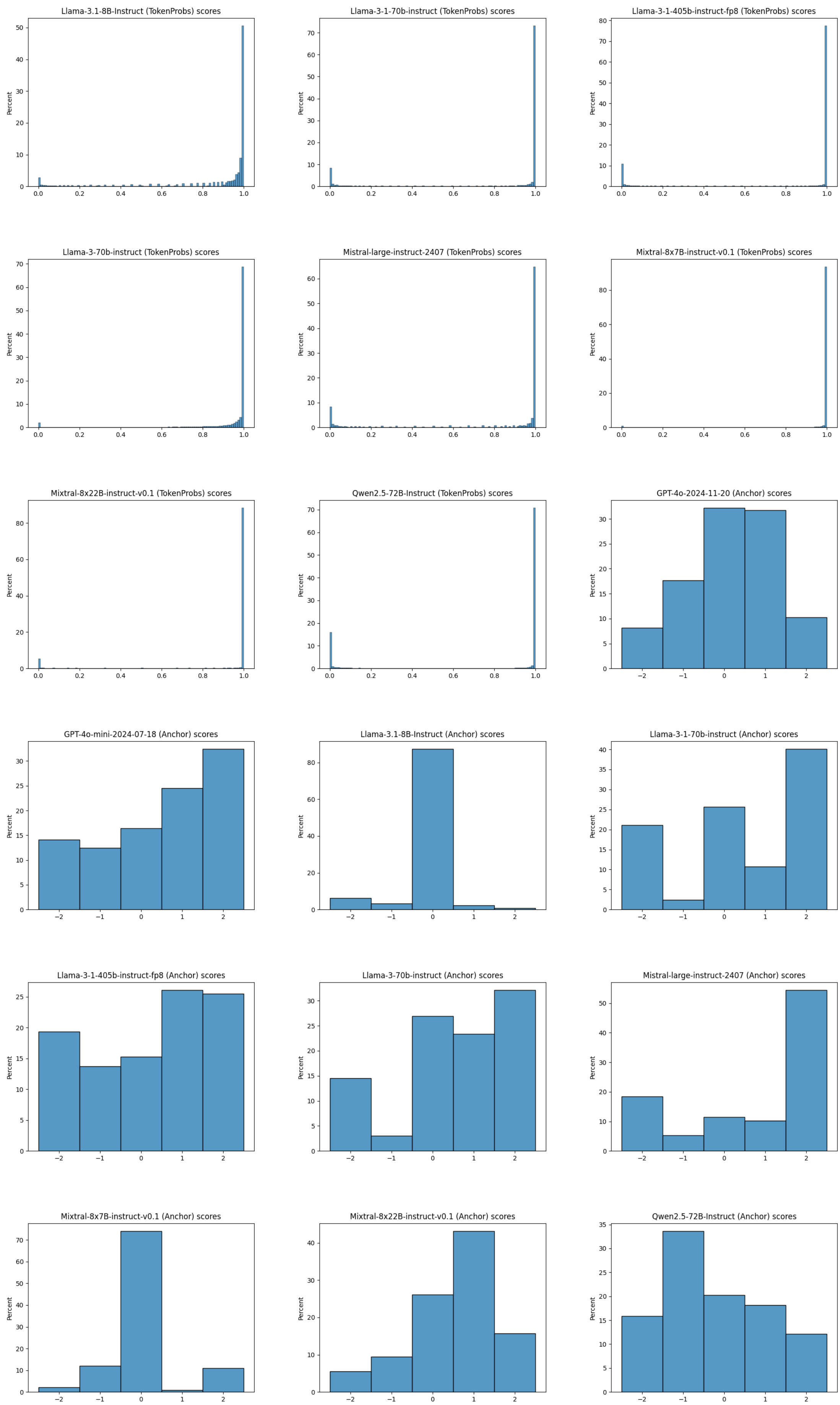


Figure 12: **Judge score distributions** *(Part 3/3)*.

Judge Model	Realization	Agreement with Gold $\tau \uparrow$	Decisiveness $\alpha \uparrow$	Bias $\delta \downarrow$
URM-LLaMa-3.1-8B	Reward	.819	1.84	.085
Qwen2.5-72B-Instruct	Likert	.817	4.76	.079
Qwen2.5-72B-Instruct	Numeric	.814	4.09	.079
Mistral-large-instruct-2407	Likert	.811	5.47	.086
GPT-4o-2024-11-20	Anchor	.809	3.07	.085
Mistral-large-instruct-2407	Numeric	.809	3.01	.082
Llama-3-1-405b-instruct-fp8	Numeric	.805	4.33	.087
GPT-4o-mini-2024-07-18	Numeric	.804	2.91	.077
GPT-4o-mini-2024-07-18	Likert	.798	4.61	.087
Llama-3-1-70b-instruct	Numeric	.798	2.69	.087
Qwen2.5-72B-Instruct	Anchor	.794	2.93	.090
Llama-3-1-405b-instruct-fp8	Likert	.787	5.22	.097
Skywork-Llama-3.1-8B-v0.2	Reward	.778	2.46	.100
Qwen2.5-72B-Instruct	TokenProbs	.777	2.69	.082
Mixtral-8x22B-instruct-v0.1	Numeric	.776	2.12	.089
GPT-4o-2024-11-20	Numeric	.774	2.15	.077
GPT-4o-2024-11-20	Likert	.773	5.49	.089
Llama-3-1-70b-instruct	TokenProbs	.765	1.26	.070
Llama-3-OffsetBias-RM-8B	Reward	.765	1.39	.076
ArmoRM-Llama3-8B-v0.1	Reward	.763	1.84	.092
GPT-4o-mini-2024-07-18	TokenProbs	.752	2.10	.084
Llama-3-70b-instruct	Numeric	.749	1.27	.084
Llama-3.1-8B-Instruct	TokenProbs	.741	.598	.061
Mixtral-8x22B-instruct-v0.1	Likert	.738	2.53	.108
Llama-3-1-405b-instruct-fp8	Anchor	.730	3.58	.112
Mistral-large-instruct-2407	Anchor	.725	2.13	.111
Llama-3.1-8B-Instruct	Likert	.723	.935	.090
Llama-3-1-70b-instruct	Likert	.722	3.90	.120
Internlm2-20b-reward	Reward	.717	1.90	.098
Internlm2-7b-reward	Reward	.712	2.35	.113
GRM-Llama3.2-3B	Reward	.711	2.30	.114
Mixtral-8x22B-instruct-v0.1	TokenProbs	.702	1.85	.088
GPT-4o-2024-11-20	TokenProbs	.700	2.22	.093
Llama-3-70b-instruct	Likert	.698	2.40	.122
Llama-3-1-70b-instruct	Anchor	.688	2.71	.126
Llama-3.1-8B-Instruct	Anchor	.677	.868	.085
Llama-3-1-405b-instruct-fp8	TokenProbs	.672	1.55	.092
Llama-3.1-8B-Instruct	Numeric	.668	1.20	.104
Llama-3-70b-instruct	TokenProbs	.663	.775	.071
GPT-4o-mini-2024-07-18	Anchor	.659	1.41	.111
Mixtral-8x7B-instruct-v0.1	Numeric	.656	1.27	.102
Mixtral-8x7B-instruct-v0.1	Anchor	.655	1.17	.102
Mixtral-8x22B-instruct-v0.1	Anchor	.641	1.50	.140
Llama-3-70b-instruct	Anchor	.633	1.82	.132
Eurus-RM-7b	Reward	.628	2.49	.138
Mixtral-8x7B-instruct-v0.1	Likert	.590	.838	.110
Mixtral-8x7B-instruct-v0.1	TokenProbs	.427	.739	.107
Mistral-large-instruct-2407	TokenProbs	.369	1.17	.123

Table 4: **Judge characteristics**. The table presents three measures for each judge realization: an overall ranking quality τ *(§5, Kendall's Tau correlation with the Chatbot Arena gold ranking)*, a decisiveness score α *(§6.1, App. F)*, and its propensity for system-specific biases δ *(§6.2)*. Correlations τ shown are for the BT aggregation method; α and δ are calculated on the judge scores before aggregation. $\downarrow$: Lower is better.