Revisiting Automatic LLM Benchers for Evaluating and Ranking Instruction Following Capabilities

Anonymous ACL submission

Abstract

Evaluating and ranking the instruction-following capabilities of different LLMs are crucial. Due to the high cost and time-consuming nature of human evaluations, an automatic LLM Bencher (i.e., a benchmark that automatically ranks the instruction-following capabilities of a set of LLMs) is always needed. An automatic LLM Bencher consists of four components: the input set, evaluation model, evaluation type, and aggregation method. However, previous work has not thoroughly explored how these components should be selected. In this paper, through controlled experiments, we provide a series of recommendations on how to choose each component. Furthermore, we found that evaluation models selected using instance-level human annotations (e.g. LLMBAR) as the ground truth do not always have better system-level evaluation capabilities, that is, as a better component of the Bencher. Lastly, we discovered that when evaluating LLMs with similar performance, the performance of the automatic LLM Bencher declines sharply.

1 Introduction

Recently, various large language models (LLMs) have been released and demonstrated strong instruction-following capabilities (Ouyang et al., 2022). Such capabilities provide an efficient way for human users to interact with LLMs and instruct them to perform various tasks, making instruction-following an important evaluation aspect of LLMs. To benchmark the instruction-following capabilities of various LLMs, human judgments of output quality remain indispensable as the gold standard, since many instruction-following tasks do not have standard answers and are open-ended (Zheng et al., 2023; Dubois et al., 2023).

Chatbot Arena (Chiang et al., 2024) embodies this concept. It is a real-time evaluation platform aimed at a large user base, where users can freely provide input, select any two hosted LLMs to generate responses, and indicate which one they prefer. Chatbot Arena periodically derives a leaderboard of various LLMs by aggregating instance-level pairwise human evaluations. To date, Chatbot Arena has collected over 1.5 million human judgments, involving more than 100 systems. Due to its substantial size and the comprehensiveness of the systems included, its LLM ranking has been widely regarded as the gold standard for developing automatic benchers of instruction-following capabilities (Li et al., 2023, 2024b; Lin et al., 2024).

However, automatic benchers are needed because of the expensive and time-consuming nature of human evaluation. As a result, several widely used automatic benchers have been proposed, such as Alpaca Eval (Li et al., 2023; Dubois et al., 2024) and Arena Hard (Li et al., 2024b), which use a strong LLM such as GPT-4 (OpenAI, 2023) to compare various systems on a carefully designed input set. These automatic benchers can produce LLM rankings that correlate well with Chatbot Arena, achieving Spearman's correlations of around 0.95. Therefore, they have been widely used to develop and evaluate LLMs for tasks like alignment fine-tuning (Tunstall et al., 2024; Yang et al., 2024).

Formally, an automatic LLM bencher for evaluating instruction following capabilities consists of the following components:

- *Input Set*: A set of instructions that each LLM is tasked with to generate responses.
- *Evaluation Model*: An LLM that serves as a proxy for human evaluators (e.g. GPT-4).
- *Evaluation Type*: The type of instance-level evaluation generated by the evaluation model, such as pointwise scores or pairwise comparisons, usually determined by the prompt.
- *Aggregation Method*: The method by which instance-level evaluations are converted into system scores and rankings.

While these automatic benchers have shown a

high level of alignment with human evaluations in Chatbot Arena, we note a lack of a systematic evaluation across all the above-mentioned components involved with the benchers. For example, while Arena Hard claims a higher alignment with human evaluation than Alpaca Eval and attributes its success to the careful selection of its input set, they also adopt different evaluation types for the evaluation: Arena Hard performs five-point pairwise comparisons, while Alpaca Eval produces binary comparisons. Therefore, the exact source of its superior performance remains unclear because of the lack of controlled comparisons.

Therefore, we aim to conduct a more rigorous examination of these automatic benchers. To this end, the first research question we explore is **RQ1: how to choose the appropriate components for building an effective automatic LLM bencher?** Specifically, we perform controlled comparisons of various input sets, evaluation models, evaluation types, and aggregation methods to investigate which choices of each component maximize the bencher's performance. Our key findings are:

(1) Input Set: Using Arena Hard as input set always yields higher correlations with the system rankings of Chatbot Arena compared to Alpaca Eval. Additionally, our efficiency analysis suggests that using 50% of the input set for the Bencher yields similar performance to using the entire input set.

(2) Evaluation Model: While proprietary models like GPT-4-turbo generally perform best as evaluation models, the open-source Llama-3.1-70B (Dubey et al., 2024) also delivers strong performance, particularly when paired with a suitable combination of other components in automatic benchers.

(3) Evaluation Type: With a fixed number of LLM queries, pointwise evaluation is slightly better than pairwise for strong LLMs, but pairwise evaluation significantly outperforms pointwise with less capable LLMs. Notably, in almost all cases, the widely used 5-point pairwise evaluation performs *worse* than the base pairwise approach.

(4) Aggregation Method: Surprisingly, when pointwise scoring is used as the evaluation type and smaller open-source LLMs serve as evaluation models, aggregating the results with the Bradley-Terry model (BRADLEY and TERRY, 1952) is far superior to simply averaging the scores.

Through the above experiments, we have identified suitable evaluation models for automatic LLM benchers, meaning we can obtain a ranking of the evaluation models based on their system-level evaluation capabilities. However, works like LLMBAR (Zeng et al., 2024) and RewardBench (Lambert et al., 2024) also ranks evaluation models based on their instance-level evaluation capabilities where the benchers are evaluated by their evaluation accuracy using instance-level human annotations instead of system rankings from Chatbot Arena as the ground truth. Thus, we aim to explore: **RQ2: can we use instance-level rankings of evaluation models as a good reference to select evaluation models for LLM benchers?** Compared to the crowd-sourced ChatBot Arena, human annotations collected for instance-level evaluations are usually of higher quality but lack the scale in terms of the number of systems used to generate evaluation examples. Sometimes system information is even unavailable for instance-level evaluation dataset. Given the significant efforts made in this direction, it would be desirable for instance-level evaluation results to also generalize to system-level evaluation, which leads to our RQ2. Our investigation of RQ2 reveals: (1) Evaluation model rankings from instance-level evaluations do *not* always align with system-level rankings, and an evaluation model that performs better at the instance level is not always better at the system level. (2) Different instance-level evaluation datasets result in evaluation model rankings with varying degrees of correlation with system-level rankings, and a more challenging dataset does not always lead to higher correlation.

Having provided a thorough evaluation of the automatic benchers, we take a step back to examine the evaluation setting itself. A critical question regarding the evaluation setting of the LLM benchers is: **RQ3: does the performance of automatic LLM benchers degrade when evaluating LLMs with similar performance?** Specifically, in RQ1 we find that we can build an effective automatic bencher using GPT-4 as evaluation models, achieving a Spearman's correlation of over 0.90 with Chatbot Arena. However, since this result is based on LLMs with varying performance levels, it does not guarantee strong bencher performance when comparing closely performing LLMs. Our exploration of RQ3 reveals that bencher performance indeed declines sharply in such cases, highlighting the limitations of existing benchers and suggesting the need for targeted evaluation in this challenging setting.

2 Problem Formalization

In this section, we formalize some crucial concepts and our research questions.

2.1 Automatic LLM Bencher

Given a set of systems to be evaluated (denoted as $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$), an automatic bencher $\mathcal{J}_A$ is composed of the following four components, which determine the rankings of these systems.

Input Set $\mathcal{X} = \{x_1, x_2, \ldots, x_m\}$. Each system s_i generates a response o_{ij} for each input x_j.

Evaluation Model $\mathcal{E}$: This is an LLM (e.g., GPT-4-turbo) that acts as a proxy for human evaluators. It generates instance-level evaluations for each system. The evaluation model $\mathcal{E}$ outputs instance-level evaluation for each response o_{ij} .

Evaluation Type $\mathcal{T}$: This refers to the type of evaluation generated at the instance level. There are two main types:

1) Pointwise Evaluation: The evaluation model assigns a score to a system's response. The pointwise score of the i-th system's response to the j-th input is denoted as $\mathcal{T}_{\text{pointwise}}(\mathcal{E}, x_j, o_{ij}) \in \mathbb{R}$. The total number of instance-level pointwise evaluations is mn.

2) Pairwise Evaluation: the evaluation model compares the quality of two responses o_{ij}, o_{kj} generated by two systems s_i and s_k on the same input x_j. The base comparison result is binary, indicating whether s_i performed better than s_k or vice versa. This can be expressed as: $\mathcal{T}_{\text{base.pairwise}}(\mathcal{E}, x_j, o_{ij}, o_{kj}) \in \{o_{ij} > o_{kj}, o_{ij} < o_{kj}\}$. In 5-point pairwise comparisons, the evaluation model provides more detailed feedback about the difference between two systems' performances, capturing both the magnitude and direction of the preference. The 5 possible comparison outcomes for systems s_i and s_k on input x_j are: $\mathcal{T}_{\text{5-point.pairwise}}(\mathcal{E}, x_j, o_{ij}, o_{kj}) \in \{o_{ij} \ll o_{kj}, o_{ij} < o_{kj}, o_{ij} = o_{kj}, o_{ij} > o_{kj}, o_{ij} \gg o_{kj}\}$. The total number of instance-level pairwise evaluations is $mn(n-1)$ rather than $mn(n-1)/2$ because we always need to swap the presentation order of the responses s_i and s_j to address the position bias of LLM-based evaluations.

Aggregation Method $\mathcal{G}$: This will create a mapping from systems to scores ($\mathcal{G}(s_i) \in \mathbb{R}$), depending on the evaluation type.

For pointwise evaluations, aggregation can be done via the mean or median of the instance-level scores across all inputs:

$$\mathcal{G}_{\text{mean}}(s_i) = \frac{1}{m} \sum_{j=1}^{m} \mathcal{T}_{\text{pointwise}}(\mathcal{E}, x_j, o_{ij})$$

$$\mathcal{G}_{\text{median}}(s_i) = \text{median}\left(\{\mathcal{T}_{\text{pointwise}}(\mathcal{E}, x_j, o_{ij})\}_{j=1}^{m}\right)$$

For pairwise evaluations, we consider the Bradley-Terry model and win ratio. They can only be applied to base pairwise instance-level evaluations, so other types of instance-level evaluations need to be converted to the base pairwise type. The conversion rules are described in Appendix A. The Bradley-Terry model $\mathcal{G}_{\text{BT}}$ computes the strength of each system via maximum likelihood estimate, and the win ratio is the proportion of pairwise wins for a system over others:

$$\mathcal{G}_{\text{win_ratio}}(s_i) = \frac{\sum_{j=1}^{m} \sum_{k \neq i} \mathbb{1}(\mathcal{T}_{\text{base.pairwise}}(\mathcal{E}, x_j, o_{ij}, o_{kj}) = o_{ij} > o_{kj})}{\sum_{j=1}^{m} \sum_{k \neq i} 1}$$

where m is the total number of inputs and the denominator counts the total number of pairwise comparisons involving s_i. Since pointwise data can be converted into pairwise data, all aggregation methods for pairwise evaluations can also be applied to pointwise evaluations after conversion.

2.2 How to Evaluate a Bencher

Ground Truth. From Chatbot Arena, we obtain system ratings, denoted as $r(s_i)$, for each system $s_i \in S$, where $r(s_i) \in \mathbb{R}$ represents the overall performance of the system as derived from human judgments. The ranking of the systems is obtained by sorting them based on their ratings: $R_H : S \rightarrow \{1, 2, \ldots, n\}$, where $R_H(s_i)$ represents the rank of system s_i based on the ratings $r(s_i)$.

Evaluation Measure. The performance of an automatic bencher $\mathcal{J}_A$ is evaluated by comparing its ranking R_A of systems with the ground truth ranking R_H, using Spearman's rank correlation coefficient ($\rho(R_A, R_H)$) and Kendall's tau ($\tau(R_A, R_H)$). ρ is more widely used.

Controllable Kendall's Tau (τ_u): Inspired by Deutsch et al. (2022), we propose controllable Kendall's tau that evaluates the agreement between rankings by focusing only on system pairs where the performance difference is small, i.e., where the absolute difference in the human-provided scores for two systems is less than a specified threshold u. This is useful for evaluating the bencher's ability to distinguish between closely matched systems.

Let $\Delta r(s_i, s_k) = |r(s_i) - r(s_k)|$ represent the performance difference between systems s_i and s_k, where $r(s_i)$ is the system ratings derived from Chatbot Arena. For a given threshold u, we define the set of system pairs that are considered for evaluation as: $P_u = \{(s_i, s_k) \mid s_i, s_k \in \mathcal{S}, \Delta r(s_i, s_k) < u,\ i \neq k\}$. The controllable Kendall's tau τ_u is then calculated only using the system pairs in P_u:

$$\tau_u(R_A, R_H) = \frac{C_u - D_u}{\sqrt{(C_u + D_u + T_{A,u})(C_u + D_u + T_{H,u})}}$$

where C_u is the number of concordant pairs (i.e., pairs of systems (s_i, s_k) where the rank order between s_i and s_k is the same in both R_A and R_H). D_u is the number of discordant pairs. $T_{A,u}$ and $T_{H,u}$ are the number of ties in the automatic bencher's ranking R_A and the human-provided ranking R_H. Only the system pairs within the set P_u are used.

2.3 Different Ways of Selecting Evaluation Models

We are comparing different evaluation models $\mathcal{E}$ based on their ability to rank systems or predict instance-level human preferences. There are three key settings for evaluating evaluation models:

Setting 1: Standard Meta-Evaluation with System-Level Rankings. The idea of Setting 1 is to rank evaluation models based on how well their corresponding automatic benchers align with the system rankings derived from Chatbot Arena. It is the standard way benchmarks such as Arena Hard are designed.

An automatic bencher $\mathcal{J}_A^{(i)}$ is composed of an evaluation model $\mathcal{E}_i$ and **fixed** input set $\mathcal{X}$, evaluation type $\mathcal{T}$, and aggregation method $\mathcal{G}$, producing a system ranking $R_A^{(i)}$. The performance of evaluation model $\mathcal{E}_i$ is measured by comparing $R_A^{(i)}$ with R_H using either Spearman's $\rho(R_A^{(i)}, R_H)$, or Kendall's $\tau(R_A^{(i)}, R_H)$ mentioned above. Evaluation models are ranked based on their performance, leading to a ranking for evaluation models: $R_{\mathcal{E}}^{(1)} : \{\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_K\} \rightarrow \{1, 2, \ldots, K\}$.

Setting 2: Instance-Level Human Judgments as Ground Truth. We consider instance-level human judgments as the ground truth, without system-level aggregation or knowledge of which systems generated the responses.

Let $\mathcal{D} = \{(x_j, o_j^{(1)}, o_j^{(2)}, h_j)\}_{j=1}^m$ represent the dataset, where each entry consists of an input x_j, two responses $o_j^{(1)}$,$o_j^{(2)}$, and human preference $h_j \in \{o_j^{(1)} > o_j^{(2)}, o_j^{(1)} < o_j^{(2)}\}$, indicating which response was preferred.

The evaluation model $\mathcal{E}_i$ generates predictions $\hat{h}_j^{(i)} = \mathcal{T}_{\text{base.pairwise}}(\mathcal{E}_i, x_j, o_j^{(1)}, o_j^{(2)})$, and the instance-level accuracy is defined as:

$$\text{Accuracy}(\mathcal{E}_i) = \frac{1}{m} \sum_{j=1}^{m} \mathbb{1}(\hat{h}_j^{(i)} = h_j)$$

Evaluation models are ranked ($R_{\mathcal{E}}^{(2)}$) based on their accuracy in predicting human preferences.

Setting 3: Instance-Level Human Judgments with System Information and Aggregation. We design this new setting to help us compare **Setting 1** and **Setting 2**. If we have access to instance-level human judgments as well as system information, we can aggregate both human judgments and evaluation model predictions into system-level rankings.

Dataset with system information: Each entry in the dataset $\mathcal{D}' = \{(x_j, o_j^{(1)}, o_j^{(2)}, s_j^{(1)}, s_j^{(2)}, h_j)\}_{j=1}^m$ includes: an input x_j, two responses $o_j^{(1)}$,$o_j^{(2)}$ generated by systems $s_j^{(1)}, s_j^{(2)}$, and human preference $h_j \in \{o_j^{(1)} > o_j^{(2)}, o_j^{(1)} < o_j^{(2)}\}$, indicating which response was preferred. Note that Chatbot Arena does not necessarily have ratings and rankings for these systems involved. The steps to producing evaluation model rankings are as follows: 1) Human preferences are aggregated to produce a system ranking R'_H with an aggregate method $\mathcal{G}$. 2) The evaluation model $\mathcal{E}_i$ provides instance-level predictions with $\mathcal{T}$, which are aggregated using the same method $\mathcal{G}$ to produce a system ranking $R_A^{(i)}$. 3) Compare the two system rankings R'_H and $R_A^{(i)}$ using Spearman's $\rho(R_A^{(i)}, R'_H)$ or Kendall's $\tau(R_A^{(i)}, R'_H)$.

The evaluation models are then ranked ($R_{\mathcal{E}}^{(3)}$) based on the correlation between their aggregated rankings and the aggregated human judgments.

Setting 1 and Setting 3 are system-level evaluation, because aggregation methods are applied and a correlation value is calculated between two set of system rankings, while **Setting 2 is the instance-level evaluation.** Note that we set the evaluation type to $\mathcal{T}_{\text{base.pairwise}}$ and aggregation method to $\mathcal{G}_{\text{BT}}$ in Setting 1 and Setting 3 to compare the results of the three settings more fairly.

2.4 Research Questions

Our research questions can be formalized as follows:

1. $\mathcal{X}, \mathcal{E}, \mathcal{T}, \mathcal{A} = \arg\max_{\mathcal{X},\mathcal{E},\mathcal{T},\mathcal{A}} \rho(R_A, R_H)$
2. Is $\rho(R_{\mathcal{E}}^{(1)}, R_{\mathcal{E}}^{(2)}) \approx 1$ and $\rho(R_{\mathcal{E}}^{(1)}, R_{\mathcal{E}}^{(3)}) \approx 1$ and $\rho(R_{\mathcal{E}}^{(2)}, R_{\mathcal{E}}^{(3)}) \approx 1$?
3. Does $\tau_u(R_A, R_H)$ decrease as u decrease?

3 Experiments and Analyses

3.1 Setups

We selected two input sets, Arena Hard (Li et al., 2024b) and Alpaca Eval (Li et al., 2023), 18 LLMs as the systems to be evaluated, and 10 LLMs, including four proprietary OpenAI models and six open-source models, to serve as evaluation models. Please see Appendix C for detailed information.

3.2 RQ1: how to choose $\mathcal{X}, \mathcal{E}, \mathcal{T}, \mathcal{A}$ to maximize the bencher's performance.

In this section, in addition to discussing how to select components for the bencher, we also conducted an efficiency analysis.

Evaluation Type. Table 1 shows the performance of a bencher under different combinations of evaluation types and aggregation methods, with several representative LLMs as evaluation models. For open-source models with fewer parameters, base pairwise is the optimal evaluation type, offering a significant advantage over other evaluation types. For proprietary models, the difference between base pairwise and pointwise evaluation types is small. Across all models used as evaluation models, 5-point pairwise generally performs worse than base pairwise, with the decline being particularly pronounced for open-source models, possibly due to their weaker ability to follow complex instructions. **Therefore, using 5-point pairwise as the evaluation type, as in Arena Hard, may be problematic.**

Aggregation Method. As shown in Table 1, when the evaluation type is base pairwise, the system rankings produced by the Bradley-Terry model and win ratio aggregation methods are mostly the same. Interestingly, **when the evaluation type is pointwise, mean, the most common aggregation method, is often not the best.** Moreover, when the evaluation model is a smaller open-source model, pointwise combined with the Bradley-Terry model significantly outperforms its combination with mean. After examination, we found that this is because the majority of instance-level scores produced by these evaluation models are very similar, and aggregating them using mean or median across all samples leads to system scores that are hardly effective. In contrast, converting these scores to base pairwise retains the slight differences in scores for different responses to the same input. On the other hand, for OpenAI models, the performance of aggregating pointwise type data using either mean or the Bradley-Terry model is comparable.

Evaluation Models. Table 2 displays the performance of benchers with different LLMs as evaluation models. We found that gpt-4-turbo and gpt-4o are the most suitable evaluation models. The rankings of evaluation models obtained from the Alpaca Eval and Arena Hard input sets show slight differences, but the correlation between them is high, with $\rho = 0.98$. Additionally, most of the smaller open-source models perform unsatisfactorily as evaluation models. However, llama-3.1-70b is an exception, performing even better than gpt-3.5-turbo and gpt-4o-mini.

Input Set. Table 1 confirms that using Arena Hard as input set always yields higher correlations with the system rankings of Chatbot Arena compared to Alpaca Eval through different combinations of $\mathcal{E}, \mathcal{T}, \mathcal{A}$.

Efficiency Analysis. In the previous analysis, we did not take into account the total number of instance-level evaluations produced by the evaluation models. In practice, when using proprietary models as evaluation models, the number of instance-level evaluations will significantly affect our costs, so we want to reduce it. According to Figure 4 in the appendix, in most cases, **using 50% of the input set for the Bencher yields similar performance to using the entire input set**, with comparable bootstrapping means and confidence intervals. Additionally, the total number of instance-level evaluations of pointwise evaluation and pairwise evaluation are different: pointwise evaluations amount to mn, whereas pairwise evaluations amount to $mn(n-1)$. Therefore, we consider it necessary to limit the number of instance-level evaluations for pairwise to be equal to that of pointwise, i.e., mn. In this scenario, when using pairwise as the evaluation type, we need to balance the **number of inputs** with the **number of comparisons per input**. Figure 1 illustrates the bootstrapping mean and confidence intervals for

Alpaca Eval as the input set									
Evaluation Model		llama-3.1-8b		llama-3.1-70b		gpt-4o		gpt-4-turbo	
Type	Aggregation	ρ	τ	ρ	τ	ρ	τ	ρ	τ
pointwise	bradley_terry	0.2982	0.1503	**0.8927**	**0.7255**	0.8782	0.7255	0.8611	0.6951
pointwise	win_ratio	0.2982	0.1503	**0.8927**	**0.7255**	0.8782	0.7255	0.8611	0.6951
pointwise	mean	0.0423	0.0719	0.2590	0.1634	0.8535	0.6993	0.8493	0.6863
pointwise	median	0.2239	0.0980	0.0815	0.0327	0.8762	0.7386	0.8803	0.7255
pairwise_base	bradley_terry	**0.5088**	**0.4118**	0.7503	0.5948	**0.8838**	**0.7475**	**0.9112**	**0.7778**
pairwise_base	win_ratio	**0.5088**	**0.4118**	0.7503	0.5948	**0.8838**	**0.7475**	**0.9112**	**0.7778**
pairwise_5point	bradley_terry	0.4056	0.2549	0.3891	0.2810	0.6615	0.5033	0.8184	0.6471
pairwise_5point	win_ratio	0.1682	0.1242	0.3560	0.2549	0.6037	0.4379	0.7977	0.6471
Arena Hard as the input set									
Evaluation Model		llama-3.1-8b		llama-3.1-70b		gpt-4o		gpt-4-turbo	
Type	Aggregation	ρ	τ	ρ	τ	ρ	τ	ρ	τ
pointwise	bradley_terry	0.5232	0.4379	0.6856	0.5246	0.9401	0.8039	0.9092	0.7778
pointwise	win_ratio	0.5232	0.4379	0.6856	0.5246	0.9401	0.8039	0.9092	0.7778
pointwise	mean	0.1744	0.0980	0.3230	0.2288	**0.9628**	**0.8693**	**0.9463**	**0.8562**
pointwise	median	0.3333	0.2941	0.5191	0.3856	0.9587	0.8562	0.9319	0.7908
pairwise_base	bradley_terry	**0.6656**	**0.5556**	**0.8741**	**0.7516**	0.9443	0.8039	0.9381	0.8431
pairwise_base	win_ratio	**0.6656**	**0.5556**	**0.8741**	**0.7516**	0.9443	0.8039	0.9381	0.8431
pairwise_5point	bradley_terry	0.3086	0.2288	0.8328	0.6863	0.9216	0.7778	**0.9463**	0.8431
pairwise_5point	win_ratio	0.2797	0.2157	0.8080	0.6471	0.9360	0.8039	0.9133	0.8039

Table 1: Standard meta-evaluation results of different combinations of input sets, evaluation models, evaluation types, and aggregation methods (the correlation between the automatic benchers' and ChatBot Arena's evaluations).

	Alpaca Eval		Arena Hard	
Evaluation Model	ρ	Ranking	ρ	Ranking
gpt-4-turbo	0.9112	1	0.9381	2
gpt-4o	0.8838	2	0.9443	1
gpt-4o-mini	0.6821	4	0.8101	4
gpt-3.5-turbo	0.4923	6	0.6636	6
llama-3.1-70b	0.7503	3	0.8741	3
llama-3.1-8b	0.5088	5	0.6656	5
glm-4-9b	0.1785	7	0.6367	7
mistral-7b-v0.3	0.1517	8	0.4076	8
mistral-7b-v0.1	0.0464	9	0.2755	10
llama-2-7b	-0.2200	10	0.2839	9

Table 2: Evaluation model rankings under Setting 1. The evaluation type and aggregation method are fixed to base pairwise and the Bradley-Terry model for all evaluation models.

base pairwise as the evaluation type under different numbers of comparisons per input. We found that:

1. When open-source models are used as evaluation models, base pairwise still has a certain advantage even when the total number of instance-level evaluations for base pairwise is limited to that of the pointwise type. However, when proprietary OpenAI models are used as evaluation models, the performance of base pairwise is always weaker than pointwise.

2. As the number of comparisons per input increases (i.e., the number of inputs decreases), both the bootstrapping mean and the lower bound of the confidence interval for bencher performance decline. **This suggests that we should prioritize using more inputs rather than conducting more comparisons per input.**

3.3 RQ2: Can we can use instance-level rankings of evaluation models as a good reference to select evaluation models for LLM benchers?

Figure 2 displays the Spearman's ρ between evaluation model rankings obtained from different datasets and settings. We found that: First, the evaluation model rankings indeed differ across different settings. Second, system-level evaluation results are more consistent with each other. The evaluation model ranking obtained from Setting 3 is very similar to that from Setting 1, despite the systems to be evaluated and input sets in Alpaca Farm being different from those in Setting 1. Third, although the input set and involved systems in `alpaca_farm_instance` and `alpaca_farm_system` are exact the same, the evaluation model rankings they produce differ substan-

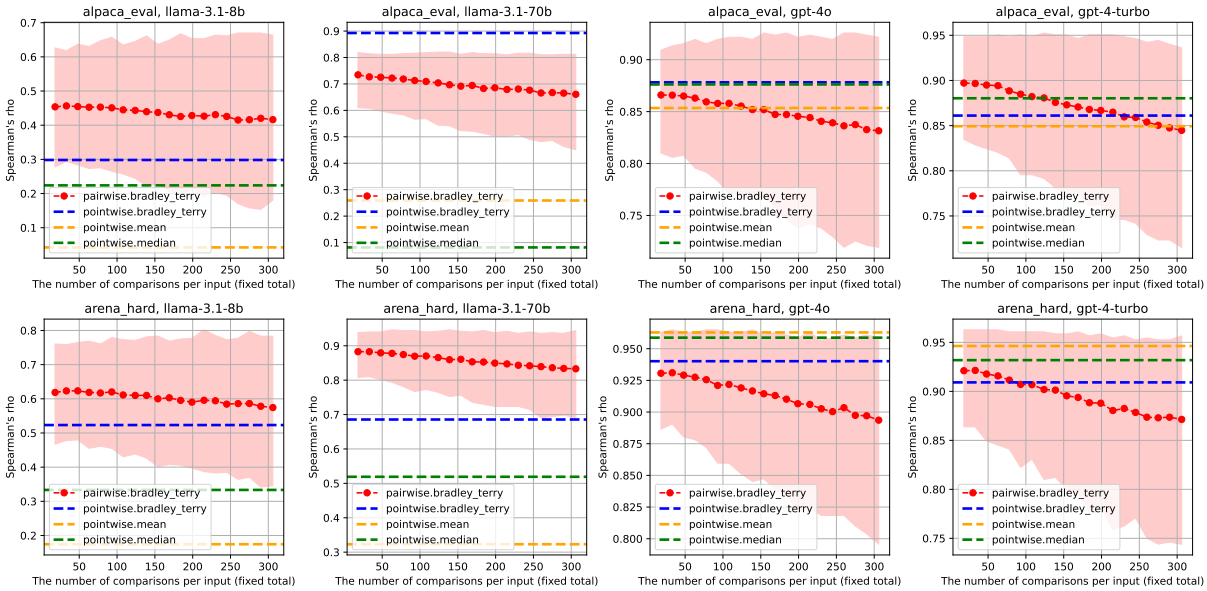

Figure 1: Bootstrapping mean and 95% confidence interval of the correlations between the system rankings of automatic pairwise evaluation and human judgment when the total number of examples is set equal to that of pointwise evaluation. The results of pointwise evaluation are depicted as horizontal lines. Both the input set and the comparisons per input are sampled with replacement. The number of iterations is set to 1000.

tially. This suggests that whether the aggregation methods are applied has a great impact on the evaluation results. Compared to `llmbar_adversarial`,

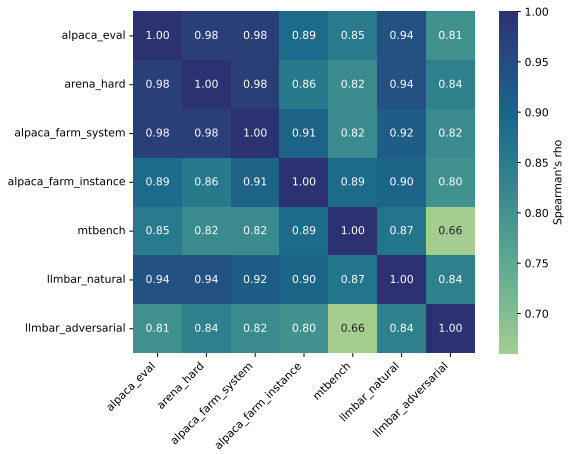

Figure 2: Spearman's ρ between the evaluation model rankings under different meta-evaluation settings. **alpaca_eval** and **arena_hard** denote the evaluation model rankings are produced under **Setting 1**. **alpaca_farm_system** denotes that evaluation model rankings are produced under **Setting 3** where the instance-level human judgment of Alpaca Farm is used as ground truth, the Bradley-Terry model is used to aggregate both automatic evaluation and human evaluation. **alpaca_farm_instance**, **mtbench**, **llmbar_natural**, and **llmbar_adversarial** refer to that the evaluation model rankings are produced under **Setting 2** where instance-level human judgment of Alpaca Farm, MT-Bench, LLMBAR are used.

the evaluation model ranking from `llmbar_natural` is more closely aligned with that from the system-level evaluation results. This is probably due to the responses in `llmbar_adversarial` are not from real systems, which makes it more different from system-level evaluation setting.

3.4 RQ3: Does the performance of automatic LLM benchers degrade when evaluating LLMs with similar performance?

As shown in Figure 5 in the appendix, the performance differences among the selected LLM systems are uneven, so the difficulty of distinguishing between different LLMs varies. We introduced a threshold u so that only system pairs with performance differences smaller than it are used to calculate τ_u. Since fewer system pairs meet the requirement as u decreases, we can control the threshold u to select a specific proportion of system pairs. Specifically, we selected 5%, 10%, ..., and 100% of system pairs and observed the changes in bencher performance. Figure 3 shows that as u decreases (i.e., the performance differences between systems become smaller), the performance of almost all benchers declines sharply. For example, when 50% of system pairs are selected, the τ_u obtained by the bencher using gpt-4-turbo as the evaluation model on Alpaca Eval drops to below 0.6.

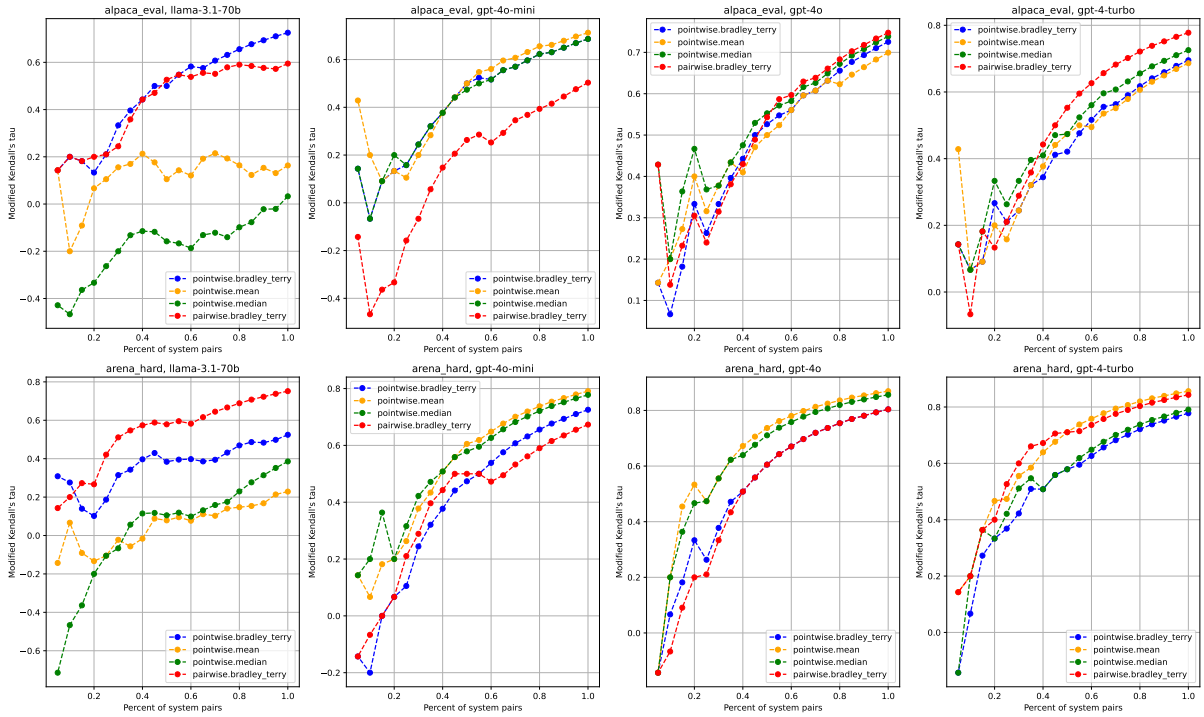

Figure 3: Controllable Kendall's tau (τ_u) between the system rankings of automatic evaluation metrics and human judgment when only partial system pairs are used. The x-axis denotes the percent of system pairs used in the evaluation, which are ascendingly ordered by their differences in the Chatbot Arena rating. Across all settings, we find that the LLM benchers' performance degrades when they evaluate close-performing systems.

4 Related Work

Automatic LLM benchers. This kind of research always create a automatic leaderboard for various LLMs. As mentioned above, the main contribution of Arena Hard (Li et al., 2024b) lies in designing a pipeline for automatically constructing a challenging input set, also for Wildbench (Lin et al., 2024). In contrast, Alpaca Eval 2 (Dubois et al., 2024), focus on designing better evaluation models and use existing instruction datasets as the input set. There is another line of research that specifically focuses on how build LLM-based evaluation models (Gao et al., 2024).

Comparison between different evaluation types and aggregation methods. Liusie et al. (2024) compared the effectiveness of LLM-based evaluation models with absolute score prediction and pairwise comparison on several NLG tasks. However, this study is not oriented to system-level evaluation and aggregation methods are not involved. Besides, we focus on the capabilities of LLMs from a wider perspective, instruction following, not limited to NLG tasks. Peyrard et al. (2021) shows the advantages of the Bradley-Terry model over mean and median on aggregating instance-level scores of NLP systems but this study focuses on what aggregation method makes more sense to choose as part of an evaluation metric. It is worth mentioning that Liu et al. (2024) investigated which evaluation type is better under the instance-level evaluation settings thoroughly but they did not explore the system-level evaluation settings.

Instance-level evaluation of evaluation models on instruction following. A lot of studies have focused on collecting instance-level human judgments to evaluate LLM-based evaluation models (Zeng et al., 2024; Zheng et al., 2023; Dubois et al., 2023; Zhang et al., 2023; Wang et al., 2024; Li et al., 2024a; Ye et al., 2024). However, to the best of our knowledge, no work has examined whether the evaluation results of them are consistent with the system-level settings.

5 Conclusions

We have explored how to construct effective LLM benchers and evaluate them. We provide several recommendations on how to combine the four components of an LLM bencher, analyze the current evaluation settings for LLM benchers and the evaluation models that serve as their components, and highlight the shortcomings in these settings.

Limitations

The observations and conclusions of this paper are dependent on the datasets and selected LLMs, and cannot be guaranteed to apply under all circumstances. Furthermore, the open-source models included as evaluation models are relatively small in scale, which may not reflect the evaluation capabilities of larger open-source models.

References

RALPH ALLAN BRADLEY and MILTON E. TERRY. 1952. RANK ANALYSIS OF INCOMPLETE BLOCK DESIGNS: THE METHOD OF PAIRED COMPARISONS. *Biometrika*, 39(3-4):324–345.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael I. Jordan, Joseph E. Gonzalez, and Ion Stoica. 2024. Chatbot arena: An open platform for evaluating llms by human preference. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

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, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 6038–6052. Association for Computational Linguistics.

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 preprint arXiv:2407.21783*.

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

Yann Dubois, Chen Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Alpacafarm: A simulation framework for methods that learn from human feedback. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Mingqi Gao, Xinyu Hu, Jie Ruan, Xiao Pu, and Xiaojun Wan. 2024. Llm-based NLG evaluation: Current status and challenges. *CoRR*, abs/2402.01383.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Raghavi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. 2024. Rewardbench: Evaluating reward models for language modeling. *CoRR*, abs/2403.13787.

Junlong Li, Shichao Sun, Weizhe Yuan, Run-Ze Fan, Hai Zhao, and Pengfei Liu. 2024a. Generative judge for evaluating alignment. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

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

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`.

Bill Yuchen Lin, Yuntian Deng, Khyathi Raghavi Chandu, Faeze Brahman, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. 2024. Wildbench: Benchmarking llms with challenging tasks from real users in the wild. *CoRR*, abs/2406.04770.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 2511–2522. Association for Computational Linguistics.

Yixin Liu, Kejian Shi, Alexander R. Fabbri, Yilun Zhao, Peifeng Wang, Chien-Sheng Wu, Shafiq Joty, and Arman Cohan. 2024. Reife: Re-evaluating instruction-following evaluation. *CoRR*, abs/2410.07069.

Adian Liusie, Potsawee Manakul, and Mark J. F. Gales. 2024. LLM comparative assessment: Zero-shot NLG evaluation through pairwise comparisons using large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2024 - Volume 1: Long Papers, St. Julian's, Malta, March 17-22, 2024*, pages 139–151. Association for Computational Linguistics.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

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

Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022.

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, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2301–2315. Association for Computational Linguistics.

Lewis Tunstall, Edward Emanuel Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro Von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M Rush, and Thomas Wolf. 2024. Zephyr: Direct distillation of LM alignment. In *First Conference on Language Modeling*.

Yidong Wang, Zhuohao Yu, Wenjin Yao, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, Wei Ye, Shikun Zhang, and Yue Zhang. 2024. Pandalm: An automatic evaluation benchmark for LLM instruction tuning optimization. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Seonghyeon Ye, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. 2024. FLASK: fine-grained language model evaluation based on alignment skill sets. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. 2024. Evaluating large language models at evaluating instruction following. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Xinghua Zhang, Bowen Yu, Haiyang Yu, Yangyu Lv, Tingwen Liu, Fei Huang, Hongbo Xu, and Yongbin Li. 2023. Wider and deeper LLM networks are fairer LLM evaluators. *CoRR*, abs/2308.01862.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. 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 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

A Conversion of 5-Point Pairwise and Pointwise Evaluation Type to Base Pairwise Type

To convert 5-point pairwise comparisons into base pairwise comparisons, we apply the following rules as Li et al. (2024b):

- $o_{ij} \gg o_{kj}$ or $o_{ij} \ll o_{kj}$ converts to 6 instances of $o_{ij} > o_{kj}$ or $o_{ij} < o_{kj}$
- $o_{ij} > o_{kj}$ or $o_{ij} < o_{kj}$ converts to 2 instances of $o_{ij} > o_{kj}$ or $o_{ij} < o_{kj}$
- $o_{ij} = o_{kj}$ converts to 1 instance of $o_{ij} > o_{kj}$ and 1 instance of $o_{ij} < o_{kj}$

For pointwise evaluation type, the conversion follows this rule, and the total number of its instances is expanded to $mn(n-1)/2$.

$$\mathcal{T}_{\text{base.pairwise}}(\mathcal{E}, x_j, o_{ij}, o_{kj}) = \begin{cases} o_{ij} > o_{kj} & \text{if } \mathcal{T}_{\text{pointwise}}(\mathcal{E}, x_j, o_{ij}) > \mathcal{T}_{\text{pointwise}}(\mathcal{E}, x_j, o_{kj}) \\ o_{ij} < o_{kj} & \text{if } \mathcal{T}_{\text{pointwise}}(\mathcal{E}, x_j, o_{ij}) < \mathcal{T}_{\text{pointwise}}(\mathcal{E}, x_j, o_{kj}) \end{cases}$$

B Definitions of Spearman's ρ and Kendall's τ

$$\rho(R_A, R_H) = 1 - \frac{6\sum_{i=1}^{n}(R_A(s_i) - R_H(s_i))^2}{n(n^2-1)}$$

Where $R_A(s_i)$ is the rank of system s_i produced by the automatic bencher and n is the total number of systems being ranked.

$$\tau(R_A, R_H) = \frac{C - D}{\sqrt{(C + D + T_A)(C + D + T_H)}}$$

Where C is the number of concordant pairs (i.e., pairs of systems (s_i, s_k) where the rank order between s_i and s_k is the same in both R_A and R_H). D is the number of discordant pairs (i.e., pairs where the rank order differs between R_A and R_H). T_A is the number of ties in the automatic bencher's ranking R_A (i.e., where two systems have the same rank). T_H is the number of ties in the human-provided ranking R_H.

C Experimental Setup

Input Set. For RQ1 and RQ3, we selected two input sets, Arena Hard (Li et al., 2024b) and Alpaca Eval (Li et al., 2023), containing 500 and 805 inputs, respectively.

Systems. 18 LLMs that ranked highly on Chatbot Arena were chosen as the systems to be evaluated[1]. The data from Chatbot Arena as of July 30, 2024, was used to compute the scores and rankings, serving as the ground truth. We reused the responses from the systems on Arena Hard, collected by Li et al. (2024b), and the responses on Alpaca Eval, collected by Li et al. (2023).

Evaluation Models. We selected 10 LLMs, including four proprietary OpenAI models and six open-source models, to serve as evaluation models. These include models gpt-4-turbo, gpt-4o, gpt-4o-mini, gpt-3.5-turbo, llama-3.1-70b-int4, llama-3.1-8b, glm-4-9b, mistral-7b-v0.3, mistral-7b-v0.1, llama-2-7b.

Evaluation Types. The evaluation types were implemented as follows. As for pointwise, given an input and a response, evaluation models were asked to score the response on a 0-9 scale. We followed the method of Liu et al. (2023), using token probability to compute a weighted sum of the model's output for the 0-9 scale as the final instance-level score. For base Pairwise or 5-Point Pairwise, given an input and two responses, evaluation models were asked to either choose the better response or make a five-category comparison. The results of the automatic evaluations were extracted using regular expressions. If the evaluation model did not provide the requested judgment, a random evaluation result was assigned to the instance. The specific prompt designs can be found in the appendix.

Aggregation Methods. We adopted Chatbot Arena's implementation of the Bradley-Terry model and implemented other aggregation methods ourselves.

Additional Setup for RQ2. We selected three datasets containing instance-level human judgments: Alpaca Farm (Dubois et al., 2023), LLMBAR (Zeng et al., 2024), and MT-bench (Zheng et al., 2023). Among these, Alpaca Farm has a larger data scale, and 22 systems are involved, making it suitable for both Setting 2 and Setting 3. The number of systems in MT-bench and LLMBAR is too small, or system information is unavailable, so they can only be used for Setting 2. Additionally, LLMBAR contains two subsets: natural and adversarial. The natural subset contains responses from real systems, while the responses of the adversarial subset were generated using adversarial techniques.

[1] gpt-4-turbo-2024-04-09, claude-3-opus-20240229, claude-3-sonnet-20240229, gpt-4-0314, gpt-4-0613, mistral-large-2402, mistral-medium, qwen1.5-72b-chat, claude-2.0, claude-2.1, gpt-3.5-turbo-0613, mixtral-8x7b-instruct-v0.1, yi-34b-chat, gemini-pro, dbrx-instruct-preview, tulu-2-dpo-70b, vicuna-33b, starling-lm-7b-alpha

D Confidence Interval Analysis

In Table 1, we computed the performance of benchers of different combinations of $\mathcal{X}, \mathcal{E}, \mathcal{T}, \mathcal{G}$, but these are merely point estimates. To assess the accuracy of our estimates, we obtained the 95% confidence intervals for bencher performance through bootstrapping. As shown in Figure 4, with an increasing number of inputs, the confidence intervals for most benchers become narrower. However, for some combinations of evaluation type and aggregation method, the confidence intervals remain relatively wide when the sample ratio equals 1.0, indicating a high degree of uncertainty in the performance estimates. In contrast, the combination of base pairwise and the Bradley-Terry model has narrower confidence intervals (the red section), suggesting that the performance estimates for this combination are more precise.

E Other Figures and Tables

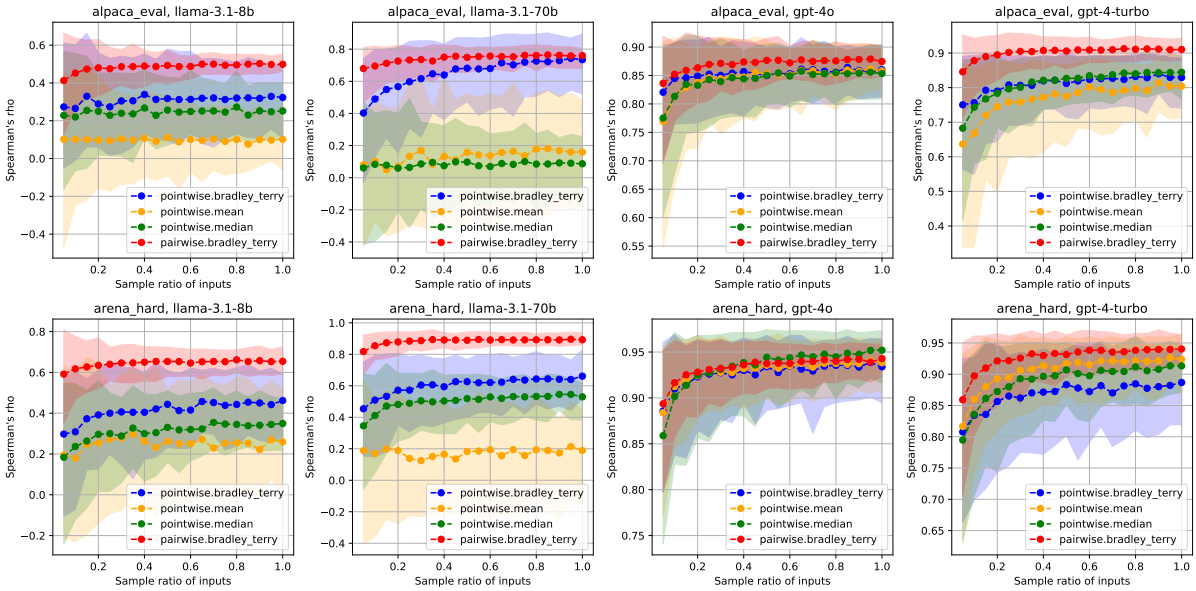

Figure 4: Bootstrapping mean and 95% confidence interval of the correlations between the system rankings of automatic evaluation metrics and human judgment. The input set is sampled with replacement. The number of iterations is set to 100.

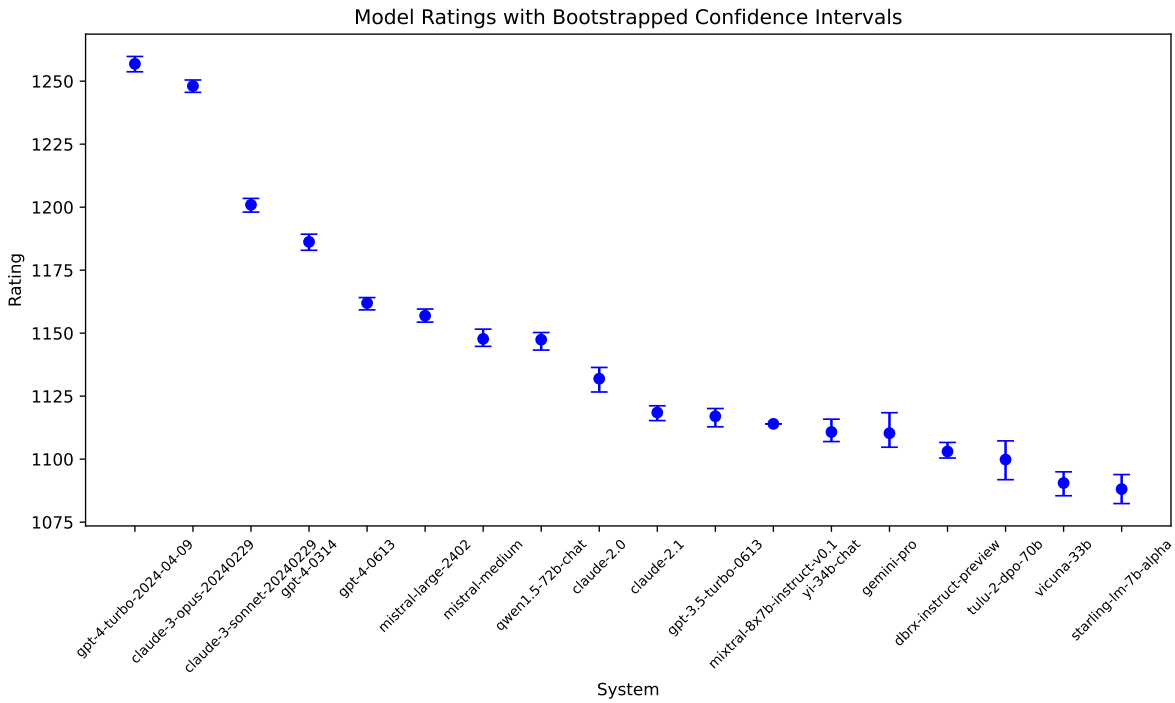

Figure 5: System rating of Chatbot Arena with bootstrapping mean and 95% confidence interval.

Prompts and Instructions
<lim_start|> system You are a helpful assistant in evaluating the quality of the outputs for a given instruction. Your goal is to score the output for the given instruction. <lim_end|> <lim_start|> user Score the Output on a scale from 0 to 9 for the given instruction, where a score of zero means "poor quality" and score of nine means "perfect quality". The output is generated by an AI chatbot. Here are some rules of the evaluation: (1) You should prioritize evaluating whether the output honestly/precisely/closely executes the instruction, then consider its helpfulness, accuracy, level of detail, harmlessness, etc. (2) Outputs should NOT contain more/less than what the instruction asks for, as such outputs do NOT precisely execute the instruction. (3) You should avoid any potential bias and your judgment should be as objective as possible. Do NOT provide any explanation for your choice. You should answer one number from 0 to 9. Do NOT output any other words. # Instruction: {INSTRUCTION} # Output: {OUTPUT} # What is your rating for the Output? <lim_end|>

Table 3: Prompt template for pointwise evaluation.

Prompts and Instructions
<lim_start|> system You are a helpful assistant in evaluating the quality of the outputs for a given instruction. Your goal is to select the best output for the given instruction. <lim_end|> <lim_start|> user Select the Output (a) or Output (b) that is better for the given instruction. The two outputs are generated by two different AI chatbots respectively. Here are some rules of the evaluation: (1) You should prioritize evaluating whether the output honestly/precisely/closely executes the instruction, then consider its helpfulness, accuracy, level of detail, harmlessness, etc. (2) Outputs should NOT contain more/less than what the instruction asks for, as such outputs do NOT precisely execute the instruction. (3) You should avoid any potential bias and your judgment should be as objective as possible. For example, the order in which the outputs were presented should NOT affect your judgment, as Output (a) and Output (b) are **equally likely** to be the better. Do NOT provide any explanation for your choice. Do NOT say both / neither are good. You should answer using ONLY "Output (a)" or "Output (b)". Do NOT output any other words. # Instruction: {INSTRUCTION} # Output (a): {OUTPUT_1} # Output (b): {OUTPUT_2} # Which is better, Output (a) or Output (b)? Your response should be either "Output (a)" or "Output (b)": <lim_end|>

Table 4: Prompt template for base pairwise evaluation.

Prompts and Instructions
<lim_start|> system You are a helpful assistant in evaluating the quality of the outputs for a given instruction. Your goal is to select the best output for the given instruction. <lim_end|> <lim_start|> user Output (a) and Output (b) are generated by two different AI chatbots for the given instruction respectively. Output one of the following choices as your verdict: 1. Output (a) is significantly better. 2. Output (a) is slightly better. 3. Tie, relatively the same. 4. Output (b) is slightly better. 5. Output (b) is significantly better. Here are some rules of the evaluation: (1) You should prioritize evaluating whether the output honestly/precisely/closely executes the instruction, then consider its helpfulness, accuracy, level of detail, harmlessness, etc. (2) Outputs should NOT contain more/less than what the instruction asks for, as such outputs do NOT precisely execute the instruction. (3) You should avoid any potential bias and your judgment should be as objective as possible. For example, the order in which the outputs were presented should NOT affect your judgment, as Output (a) and Output (b) are **equally likely** to be the better. Do NOT provide any explanation for your choice. Do NOT say both / neither are good. You should answer using ONLY 1, 2, 3, 4, or 5. Do NOT output any other words. # Instruction: {INSTRUCTION} # Output (a): {OUTPUT_1} # Output (b): {OUTPUT_2} # What is your verdict? Your response should be 1, 2, 3, 4, or 5: <lim_end|>

Table 5: Prompt template for 5-point pairwise evaluation.