Rating Roulette: Self-Inconsistency in LLM-As-A-Judge Frameworks

Rajarshi Haldar and Julia Hockenmaier
University of Illinois Urbana-Champaign
{rhaldar2, juliahmr}@illinois.edu

Abstract

As Natural Language Generation (NLG) continues to be widely adopted, properly assessing it has become quite difficult. Lately, using large language models (LLMs) for evaluating these generations has gained traction, as they tend to align more closely with human preferences than conventional n-gram or embedding-based metrics. In our experiments, we show that LLM judges have low intra-rater reliability in their assigned scores across different runs. This variance makes their ratings inconsistent, almost arbitrary in the worst case, making it difficult to measure how good their judgments actually are. We quantify this inconsistency across different NLG tasks and benchmarks and see if judicious use of LLM judges can still be useful following proper guidelines.

1 Introduction

As Natural Language Generation (NLG) becomes more prevalent in a wide variety of applications like automated journalism, customer service chatbots, language translation and content summarization, proper evaluation and measurement of alignment with human preference are critical to improve user trust and system utility. Although traditionally used automatic metrics like BLEU (Papineni et al., 2002) and BERTScore (Zhang* et al., 2020) work well with tasks like translation when multiple references are available, they fail to work in more open-ended tasks like summarization or general chatbot settings.

To automatically evaluate LLMs in these settings, LLM-as-a-judge (Li et al., 2024; Gu et al., 2025) has emerged as an automatic, scalable alternative to manual evaluation. The common practice is to prompt an LLM to evaluate natural language generations. The ratings obtained are then verified by comparing with ratings assigned by human judges, which are taken as the gold standard. This comparison is commonly done through computing exact match for categorical labels or correlation for numeric or ordinal scales (Liu et al., 2023; Thakur et al., 2025a).

However, what is often missing from these studies is any measure of self-reliability or intra-rater reliability of both LLM and human judges. We define self-reliability as the agreement of a judge with itself over multiple runs with the same settings (prompt and hyperparameters in case of LLMs).

Meanwhile, self-reliability data is completely missing from human annotations in existing benchmarks. In some cases, annotations from multiple human judges exist, but they only give us information about the inter-rater reliability instead, which is also frequently lower than the conventionally agreed upon agreement thresholds (Falke et al., 2019; Fabbri et al., 2021; Pagnoni et al., 2021). Meanwhile, agreement between LLM and human judges is usually computed using metrics like correlation and accuracy, instead of using metrics specifically designed for measuring agreement like Krippendorff's Alpha. This can often lead to an overestimation of agreement since those metrics do not account for chance agreement (Thakur et al., 2025b).

NLG can be evaluated using different metrics on a variety of tasks such as **machine translation**, **dialog generation** and **summarization**. In our experiments, we first study the simplest case of evaluating summarization — assigning a **binary label** to a summary in the **SummaC** benchmark (Laban et al., 2022). Next, we look at evaluating summaries in more complex scenarios, using a **Likert rating scale** of 1 to 5 to rate a summary on several metrics like coherence, consistency, fluency and relevance. We can use the **SummEval** benchmark for this purpose. Finally, we look at evaluating a different task of NLG, where we use a judge LLM to **rank** multi-turn conversations from two competing LLMs in the **MT-bench** benchmark (Zheng et al., 2023).

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 24986–25004
November 4-9, 2025

In this work, we make several key contributions to understanding the reliability of LLM-generated ratings in NLG evaluation. First, show that ratings output by LLMs have low agreement over multiple runs with the same prompt. Second, we show that turning off any sampling to make an LLM always output the same rating hurts performance measured by agreement with human judgment. Third, we find that this phenomenon persists across multiple tasks and benchmarks related to NLG, indicating that this is a widespread problem that needs to be addressed. Finally, we discuss some recommendations to conduct more robust NLG evaluations.

2 Background

Evaluating Natural Language Generation The gold standard for NLG evaluation has relied on **human-centric evaluations**, where human judges assess generated text. However, this is time and cost-intensive, prone to unreliability and biases (Thomson et al., 2024), and existing studies report low inter-annotator agreement or omit them altogether. For example, Celikyilmaz et al. (2021) found that only 18% of 135 surveyed papers included agreement analysis. Furthermore, leaderboards like Chatbot Arena (Chiang et al., 2024) employing crowdsourced evaluations have inequities leading to an uneven playing field (Singh et al., 2025). Unlike human evaluations, automatic metrics like BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), and METEOR (Banerjee and Lavie, 2005), and model-based metrics like BERTScore (Zhang* et al., 2020) and BLEURT (Sellam et al., 2020) are faster, less subjective and more scalable. However, these metrics often fail to fully capture human notions of quality, especially for subjective tasks.

LLM-as-a-judge Large language models can serve as judges by assigning scores, rankings, or labels to generated outputs, a paradigm surveyed in recent works (Li et al., 2024; Gu et al., 2025). LLM judges have been applied across 20 NLP tasks (Bavaresco et al., 2024) and integrated into AI-assisted human evaluation (Ashktorab et al., 2024). Notably, GPT-4 with chain-of-thought prompts aligns more closely with human judgments than conventional automatic metrics on NLG tasks (Liu et al., 2023; OpenAI et al., 2024; Wei et al., 2022). Evaluation typically involves comparing LLM outputs to human judgments via correlation (Liu et al., 2023; Xiao et al., 2023; Bavaresco et al., 2024) or percentage agreement (Zheng et al., 2023) on benchmarks such as JudgeBench (Tan et al., 2024). Challenges include bias (Zheng et al., 2023; Li et al., 2025), uncertainty in judgments (Wagner et al., 2024), adversarial vulnerability (Gu et al., 2025), and domain-specific performance gaps (Tseng et al., 2024), in addition to existing problems of LLM generations like prompt sensitivity (Mizrahi et al., 2024). To enhance reliability, methods like panel-based evaluation (PoLL) (Verga et al., 2024) and comparative studies of fine-tuning versus GPT-4 prompting (Huang et al., 2024; OpenAI et al., 2024) have been proposed. There is another challenge, high variability in LLM outputs leading to inconsistencies in LLM judges.

Variability in LLM Judgments Due to the stochastic nature of LLMs, they give different outputs when run on the same prompt multiple times. While this is by design, it should not alter the ratings assigned by an LLM judge to the same rating. Chiang and Lee (2023) observed some variability in ratings produced by LLM judges on the WritingPrompts (Fan et al., 2018) dataset. Liu et al. (2023) took advantage of this variability in GPT-4 by setting temperature to 1 and sampling 20 times to get multiple scores and then getting the final rating by normalizing all the scores by their probabilities. In our work, we go further to analyze the implications of high variability in assigned judgments, discuss metrics to measure this variability, study the extent of this issue across different tasks and benchmarks, and explore whether we would get better results by turning off the variability by setting the temperature to zero. We frame this variability as **self-reliability** or **intra-rater reliability**.

Self-reliability Also called intra-rater reliability, this measures the consistency of a single evaluator's judgments across repeated assessments, as opposed to inter-rater reliability. Common metrics include the intraclass correlation coefficient (ICC) for continuous scales (Koo and Li, 2016; Bent et al., 2014), Cohen's Kappa for categorical ratings, and Krippendorff's Alpha for ordinal, interval, or ratio data with missing values (Krippendorff, 2011; Wiethölter et al., 2023). Reporting intra-rater reliability is standard in domains like essay grading (Cohen, 2017), physical therapy (Mischiati et al., 2015), and clinical diagnostics (Królikowska et al., 2023), and its joint consideration with inter-rater reliability is urged by Harvey (2021). Yet, NLP annotation

and LLM-evaluation studies frequently omit self-reliability metrics (Abercrombie et al., 2023a,b), a deficiency we confirm in our LLM-as-a-judge experiments, where models exhibit unstable self-judgments.

3 Experiments

We first consider a simple framework for NLG evaluation, binary classification of machine-generated summaries. A good candidate for this is **SummaC**, a factual consistency detection benchmark.

Following this, we make a deeper dive into the performance of LLM judges on the **SummEval** benchmark (Fabbri et al., 2021). Compared to the binary labels in the SummaC benchmark, the summaries in this benchmark are evaluated by multiple human raters on a Likert scale across multiple metrics.

Zheng et al. (2023) introduced the MT-bench benchmark, which evaluates an LLM's multi-turn conversational and instruction-following ability. While related to NLG, it is a very different task compared to summarization.

3.1 Datasets

Evaluating A Summary From SummaC

Article

Migaloo is known for his distinctive colouring and for many years was the only documented all-white humpback whale in the world. He has been sighted off the coast of New South Wales state, including the resort town of Byron Bay. Migaloo's journey up Australia's east coast has attracted large numbers of whale enthusiasts. The 14m-long mammal was spotted with a companion during his venture north but now appears to be travelling solo. A Twitter account run by the White Whale Research Centre provides real-time updates of the whale's whereabouts.

Summary

A humpback whale spotted off the coast of Australia has been captured off the coast of Western Isles, scientists have said.

Label

1 (inconsistent)

Figure 1: Annotating a summary of an article in SummaC with 1 (inconsistent).

SummaC The SummaC benchmark (Laban et al., 2022) evaluates factual consistency in text summarization by requiring judges to label summaries as either consistent or inconsistent with their source documents (example in Figure 1). It unifies six datasets, CoGenSumm (Falke et al., 2019), XSumFaith (Maynez et al., 2020), Polytope (Huang et al., 2020), FactCC (Kryscinski et al., 2020), SummEval (Fabbri et al., 2021), and FRANK (Pagnoni et al., 2021), into a binary classification task. Standard validation/test splits are used where available; otherwise, splits are created as needed. It is licensed under the Apache License Version 2.0. Dataset statistics are provided in Table 4 in Appendix C.1.

Evaluating a Summary from SummEval on Four Metrics

Article

Manchester City are keen to sign Anderlecht teenager Evangelos Patoulidis. The 14-year-old playmaker is regarded as one of the best talents to emerge from Anderlecht's youth set-up and has also attracted attention from Arsenal and Barcelona. The Belgian starlet rejected a move to Barcelona's La Masia academy when he was 12 as his family wanted him to continue his studies. He has continued to impress and City have held discussions with Anderlecht chairman Roger Vanden Stock in the hope of agreeing a compensation package. Manuel Pellegrini is looked to build for the future by snapping up hot property Evangelos Patoulidis.

Summary

Evangelos Patoulidis is regarded as one of the best players to emerge from Anderlecht youth. He has also attracted attention from Arsenal and Barcelona. The Belgian starlet rejected a move to Barcelona's La Masia academy. The 14-year-old has attracted interest from Barcelona to Barcelona.

Annotation Scores

Coherence:	3
Consistency:	5
Fluency:	4
Relevance:	3

Figure 2: Annotating a summary from the SummEval benchmark with scores ranging from 1 to 5 on the metrics: coherence, consistency, fluency and relevance.

SummEval This is a summarization benchmark with 1700 examples where judges rate model-generated summaries of source documents on a 1–5 scale across four metrics: **coherence** (how well the sentences in the summary fit together), **consistency** (the factual accuracy of the summary), **fluency** (grammatical correctness and stylistic quality of each sentence in the summary), and **relevance** (whether the summary accurately captures the article's main points without including unnecessary details). It is licensed under the MIT license. Each example includes scores from both 3 expert and 5 crowd-sourced annotators. The authors presenting this benchmark reported some agreement metrics. They found the inter-annotator interval kappa to be below an acceptable range, 0.492 and 0.413 for the crowd-sourced workers and the first round of expert annotations accordingly. However, the second round of expert annotations improved the inter-annotator agreement achieving a kappa coefficient of 0.7127.

MT-Bench In a multi-turn conversation from this benchmark, a user prompts two LLMs with a ques-

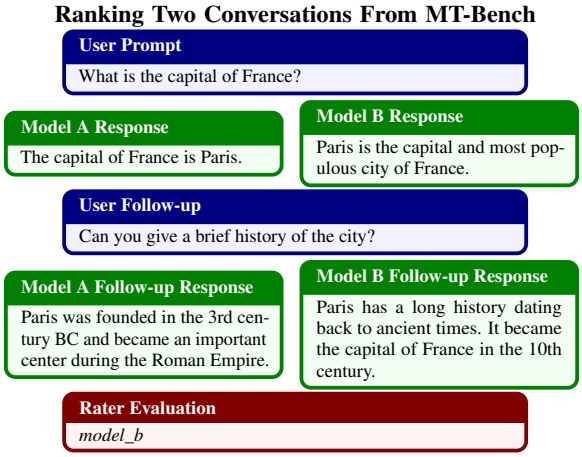


Figure 3: Ranking conversations from MT-bench with by indicating *model_a*, *model_b* or *tie*.

tion, and after receiving their responses, asks a follow-up question, to which a second pair of responses is generated. A rater (human or LLM) is shown this conversation and has to assign a label (*model_a*, *model_b*, *tie*) indicating model preference. Figure 3 shows an example of a conversation from this dataset getting rated.

There are 80 questions that require an LLM to perform multi-turn conversations and follow instructions on topics including reasoning and math. There are 30 examples for each question, where each example in the dataset comprises a question followed by responses from two models, *model_a* and *model_b*, for a total of 2.4k examples. Each example also contains a judgment assigned by GPT-4 and judgments from zero to five human raters. Since we cannot perform any agreement analysis between human raters for fewer than 2 raters, we create a smaller filtered subset of the data containing 761 examples where each example contains two or more human ratings. Table 7 in Appendix E.1 shows the distribution of the number of human ratings in the dataset that we use in our experiments.

3.2 Experimental Settings

For all three benchmarks, three Large Language Models (LLMs) are used as judges. We use the following models across all benchmarks: Llama-3.1-70B-Instruct (Grattafiori et al., 2024), DeepSeek-R1-Distill-Qwen (DeepSeek-AI et al., 2025), and Qwen3-32B (Yang et al., 2025). Details of the hyperparameter configurations and prompt templates are available in Appendix A.

For all three benchmarks, we ran each judge LLM on the same set of generations independently for three runs. We used the same prompts and settings for each run to measure intra-rater variance under fixed conditions. For **SummaC**, we added the articles and the corresponding generated summaries from the test set to be rated by each model to the prompt. Once we had three sets of scores for a benchmark and for an LLM judge we computed intra-rater reliability using Krippendorff's Alpha. In our initial experiments, we also tried running our LLM judges on additional runs (up to 10) but found no significant effect of the number of runs on self-reliability so ultimately, we kept the number of runs to 3.

SummEval allows reliability analysis along specific evaluation metrics: Coherence, Consistency, Fluency, and Relevance. We prompted each LLM judge to evaluate each metric on a scale of 1 to 5 independently per run. That means for each of the three runs we prompted the LLM four times, once for each metric. To better align with ordinal rating behavior, we replace SummEval's default interval-based distance metric with ordinal distance for computing agreement, as the ratings are on a 1–5 Likert scale. This allows a more principled estimate of intra-rater reliability by accounting for ordinal semantics.

For **MTBench**, we closely follow the setup from Zheng et al. (2023), with the addition of measuring judge consistency when evaluating chatbot responses over three runs per LLM judge. Each LLM judge scores the same prompts using the same interface scripts as the original benchmark. In addition to intra-rater reliability, we also analyze human-human and human-LLM agreement using the expert and crowd-sourced annotations provided in the benchmark, with human judgments from GPT-4 used as a reference point for LLM-human comparisons.

The prompts used in each of these benchmarks are in Appendix.

3.3 Agreement Metric

As we have multiple independent ratings from an LLM judge on a single item, we can adapt existing agreement metrics to measure self-reliability of an LLM judge. For both self-reliability of an LLM judge and its agreement with other judges, we primarily use **Krippendorff's Alpha**. We describe this metric and the choice of distance functions in Appendix B. Additionally, we also use this metric to report inter-rater agreement across categories of judges (e.g., LLM vs. human, or expert vs. crowd).

Why Krippendorff's Alpha Other works studying LLM-as-a-judge typically use different metrics for computing agreement between LLM and human judges. For example, G-Eval (Liu et al., 2023) used correlation and MTBench (Zheng et al., 2023) used accuracy. So why go with a different metric? Artstein and Poesio (2008) discussed several drawbacks when metrics without chance-correction like accuracy and correlation. Despite being intuitive to understand, their values cannot be compared across studies, because some of the agreement will be due to chance, and that chance is affected by factors that vary from one study to another. One factor is that chance agreement is higher when there are fewer categories or labels. For example, in binary classification with uniformly distributed labels, the chance agreement will be 50%, whereas when we have three labels it will be 33%. Another reason these metrics cannot be trusted is that they do not correct for for the distribution of items across categories. For example, in binary classification, if one label appears 95% of the time, the chance agreement for that label will be $0.95 \times 0.95 = 0.9025$, and it will be $0.05 \times 0.05 = 0.0025$ for the other label. This would mean that two raters would be expected to agree 90.5% of the time, and an observed agreement of 90% may look high but is actually worse than what we would expect to get by chance. A chance-corrected metric like Krippendorff's Alpha addresses these drawbacks. It also has advantages over other chance-corrected metrics like Cohen's Kappa (Cohen, 1960) and Fleiss' Kappa (Fleiss, 1971) with more flexibility supporting varying numbers of annotators, handling missing data, and accommodating different distance functions suited to the rating scale of each benchmark. Due to these reasons, we believe Krippendorff's Alpha is appropriate for measuring both self-reliability in LLM judges and their agreement with human judges.

We adapt the underlying distance function for Krippendorff's Alpha based on the nature of the ratings in each benchmark. In **SummaC**, labels are binary and categorical, so we use *nominal distance*, which treats all disagreements equally without assuming an ordering. We also report **Balanced Accuracy**—the mean of sensitivity and specificity—when comparing LLM and human judgments, following the benchmark's original evaluation setup and accounting for class imbalance.

In **SummEval**, ratings are ordinal, so we use *ordinal distance* to reflect varying degrees of disagreement (e.g., a 1-point difference counts less than a 3-point difference). Agreement is computed between cross-category judge pairs only, excluding intra-category comparisons.

Model	Self-Reliability
Llama 3.1	0.3263
Deepseek-R1	0.6278
Qwen-3	0.7883

Table 1: Self-reliability of different LLM judges on the SummaC benchmark across 3 runs measured by Krippendorff's Alpha.

For **MTBench**, we use two metrics: (1) **Accuracy**, matching the original benchmark, measures agreement with the gold standard (majority human vote); (2) Krippendorff's Alpha with *ordinal distance* captures gradations in pairwise disagreement (e.g., *model_a* vs. *tie* vs. *model_b*). Both inter-rater (e.g., LLM vs. human) and intra-rater (across LLM runs) agreement are reported.

4 Intra-Rater Reliability Results

SummaC Table 1 shows the Krippendorff's Alpha computed for LLM judges over 3 runs. We see that the agreement value is low for all models for all runs, though it gets better for newer and larger models, with Qwen 3 getting close to 0.8, which is the commonly accepted threshold of good agreement. We tried repeating these experiments for up to 5 runs but found no significant difference in self-reliability, suggesting that this value is fairly static for a model on a specific test set, independent of the number of runs.

SummEval In Figure 4, we see that compared to human evaluators, the different runs in Deepseek-R1 and Qwen 3 show a high self-reliability on Coherence and Consistency, with very low self-reliability on Fluency. Not only does Fluency have low reliability for all our LLM judges, but it is also the only metric where Llama performs best. This suggests that an LLM judge's ability to reliably produce ratings depends heavily on the metric.

MTBench We found that the self-reliability of LLM raters is much lower on this benchmark compared to SummaC and SummEval. Llama 3.1 had a Krippendorff's Alpha of **0.265** across its 3 runs, while for Deepseek-R1 it was **0.507**. Qwen 3 was the most reliable with a Krippendorff's Alpha of **0.563**. Even in the best case, these numbers are

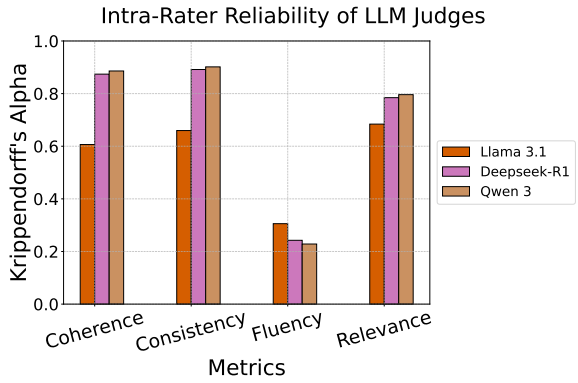


Figure 4: Self-reliability of LLM judges on SummEval.

much lower than the desired threshold of 0.8, showing that LLM raters are extremely volatile on this task, even more so than summarization. In fact, Qwen 3 gave the same judgment on all 3 runs for only 61.3% of cases.

5 Are LLM Judges a Reliable Substitute for Human Judges?

Our next set of experiments studies what this lack of self-reliability implies in terms of real-world performance, which for benchmarks usually means comparing with human annotations. In **SummaC**, we measure the agreement of LLM judgments with human annotations, while in **SummEval** and **MT-Bench**, we have multiple human labels available per example, allowing us to compare the agreement between LLM judges and human judges with human-human agreement.

5.1 SummaC

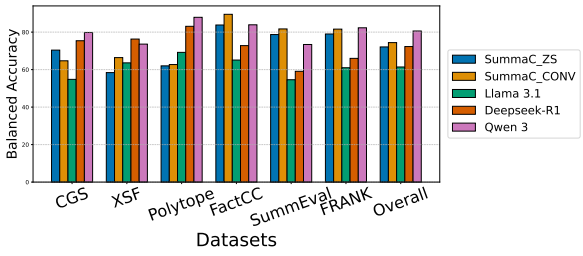


Figure 5: Balanced Accuracies of different LLM judges against human judgments on the datasets in SummaC.

In Figure 5, we report the balanced accuracies of our three LLM judges on the different datasets within SummaC, along with the two baselines, $SummaC_{ZS}$ and $SummaC_{CONV}$ that were introduced by Laban et al. (2022). Since we had 3 runs for each LLM judge, we compute the consensus rating between the 3 runs by taking a simple majority vote and then find the agreement between it and the human label. We see that while Llama 3.1 has very low performance, Deepseek-R1 is competitive with the baselines and Qwen 3 significantly outperforms the baseline models overall. It is to be noted that the baselines $SummaC_{ZS}$ and $SummaC_{CONV}$ were fine-tuned on this task whereas in our experiments we prompted the instruction-tuned models off-the-shelf.

Model	Single Run (Mean ± Std)	Majority	No Sampling
Llama 3.1	59.1 ± 2.06	61.4	58.4
Deepseek-R1	69.8 ± 0.50	72.3	69.3
Qwen 3	79.4 ± 0.32	80.6	79.2

Table 2: Balanced accuracies of LLM judgments against human judgments under different experimental settings.

Despite performing best overall, Qwen 3 is not the best at all datasets in the benchmark. Deepseek-R1 performs the best at XSumFaith. $SummaC_{CONV}$ performs the best at FactCC, which is unsurprising since that baseline was specifically fine-tuned on training examples from that dataset (Laban et al., 2022). Therefore, how well an LLM can substitute a human judge varies greatly on the dataset in question, though in general, newer and larger models perform better.

In Table 2, we see that accuracy varies between runs much more for Llama 3.1, while it is relatively stable for Deepseek-R1 and Qwen 3.

Does Taking Majority Vote Help Performance? In Table 2, for each LLM, we report the balanced accuracy of a single run, the accuracy we get if we take the consensus rating of the three runs instead, as well as the accuracy if we turn off sampling to ensure we get the same output every time. We see that accuracy is higher when computed via majority vote than the expected accuracy for a single run. In fact, for Deepseek-R1 and Qwen 3, taking the majority vote gives a higher balanced accuracy than the maximum accuracy achieved by a single run. Not only does running the LLM multiple runs accounts for this variance between runs, but it also gives higher performance in terms of accuracy against human judgments. This performance is also competitive with the baselines $SummaC_{ZS}$ and $SummaC_{CONV}$ (Laban et al., 2022), which were fine-tuned specifically for this task.

Does disabling temperature sampling reduce variance without degrading performance? The main reason this issue exists is that these LLMs are

designed to perform sampling to generate slightly different responses to the same prompt. Therefore, to remove variance in ratings, the most obvious answer is to run inference without any sampling to get the same output every time. However, in Table 2, we see that for both models, there is a degradation in performance if run without sampling. This shows that there is a trade-off between self-reliability and performance, and it is not trivial to address both issues simultaneously. We also ran additional experiments in Appendix C.2 to see if incorporating few-shot or chain-of-thought prompting led to any reliability or agreement gains and found that to not be the case.

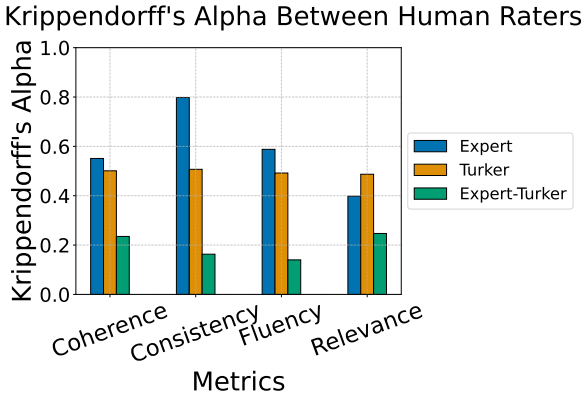


Figure 6: Inter-rater reliability within and across both categories of human judges on SummEval.

5.2 SummEval

Figure 6 presents Krippendorff's Alpha agreement scores among human annotators. Experts show the highest agreement on consistency (0.798), moderate agreement on fluency (0.588), and the lowest on relevance (0.398), suggesting subjectivity in assessing relevance. In contrast, crowdworkers (Turkers) display uniformly lower agreement across all metrics (0.48–0.51), indicating more limited task comprehension. Agreement between experts and Turkers is drastically lower (maximum 0.247 for relevance), highlighting fundamental differences in the evaluation approaches of different populations of raters.

While LLM judges show higher agreement with themselves than human judges do with each other, this does not guarantee they can fully replace human judges. In Figure 7, we see that LLMs' agreement with human experts varies significantly depending on the metric, dataset, and the expertise of the human judge.

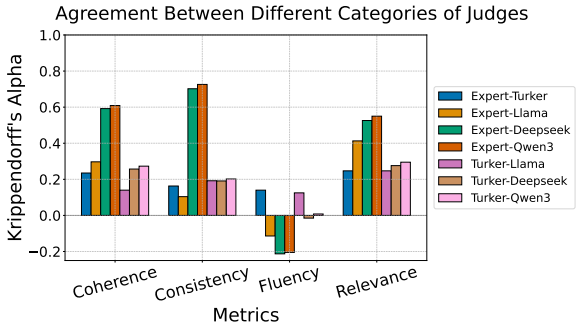


Figure 7: Inter-rater reliability of LLM judges against both types of human judgments on SummEval.

For instance, Turker ratings diverge from both experts and LLMs, indicating they use conflicting evaluation criteria. Experts and LLMs show somewhat higher, but still modest agreement, especially on metrics like consistency and coherence, while agreement drops or even turns negative for subjective metrics such as fluency.

Model scaling (e.g., using larger models like Qwen 3) can improve agreement with experts for some metrics, but even the best observed agreement (0.726 for consistency) remains below the commonly accepted threshold for substitution. Moreover, score distributions differ across judge types and metrics, as reported in Appendix D.1, further complicating direct replacement.

5.3 MTBench

We observe a fairly low Krippendorff's Alpha of **0.478** between human judges. This follows from our results under SummEval, where we find that crowdsourced judges tend to have very lower agreement. However, accuracy is comparatively much higher at **0.827**, suggesting that it may be a misleading metric when showing agreement between multiple judges, since it does not take the probability of chance agreement into account, leading to inflated scores. A complete breakdown of agreements of different judges against humans is shown in Table 3.

Table 3 shows the agreement between different judge types on the MT-bench benchmark.

Agreement between LLMs and humans Similar to agreement between humans, we see that when we use Krippendorff's alpha, we get much lower values of agreement than accuracy. We see that GPT-4 has higher values of agreement across both metrics, though it is still much lower than between humans. However, even for humans this is far lower

Judges (Human vs.)	Accuracy	Krippendorff's Alpha
Human	0.827	0.478
GPT-4	0.671	0.396
Llama 3.1	0.556	0.239
Deepseek-R1	0.668	0.385
Qwen 3	0.719	0.426

Table 3: Agreement of different judges against humans.

than the accepted threshold of 0.8, and is much lower than the agreement we saw for SummEval, suggesting this task is even harder to consistently than summarization.

6 Discussion

6.1 Takeaways

SummaC Even though simply prompting an LLM to score a generation can perform comparable to models specifically fine-tuned for this task (Laban et al., 2022), it calls into question what high performance means in this context when there is high variance between runs. For example, what does an agreement of 0.8 between LLMs and human judges mean when the LLM has low agreement with itself? We also do not know if this phenomenon was due to the limitations of the benchmark itself (e.g. noisy examples or a poorly defined task) or is more widespread across other benchmarks or tasks. More importantly, we do not know if human judgments, which are taken as the gold standard, suffer from the same consistency issues as LLMs, which should prompt further investigations. However, SummaC contains only one human judgment per example, making such investigations impossible, so we must look to other benchmarks and tasks. We saw the LLM judges struggle particularly with one of the datasets within this benchmark, SummEval (Fabbri et al., 2021). In this dataset, we also have multiple human judgments for each example across multiple metrics, which would allow us to address a lot of these questions that so far remained unanswered.

SummEval We see that the problem of low intra-rater reliability of LLM judges is not just due to binary labels in the SummaC benchmark but also persists across other metrics and scoring scales. Not only does it make LLM judges unreliable, but it also makes meta-evaluation of different evaluation frameworks difficult when the scores assigned by the same judge fluctuate over time. This raises another important question, is the low intra-rater reliability due to limitations of the LLMs or is the task of summarization not well-defined even with different metrics? It is possible that scoring the summaries in the SummEval benchmark along metrics like coherence or consistency is not a well-defined task, which causes the scores to fluctuate for both human and LLM judges. We need to study whether this phenomenon persists for benchmarks in other tasks.

MT-Bench This confirms that it is not just single-turn tasks like summarization where intra-rater reliability is an issue. It is a widespread phenomenon across benchmarks as well as tasks that involve text generation. We also observe that metrics like accuracy inflate agreement values compared to metrics like Krippendorff's Alpha.

6.2 Recommendations

Based on these findings, we can make the following recommendations to improve the state of NLG evaluation with LLM as well as human judges.

Account For Possible Self-Reliability Issues Future work should incorporate intra-rater reliability information into evaluation frameworks and explore methods to improve LLM self-reliability without sacrificing agreement with human judgments. Existing NLG research already uses metrics like **Cohen's Kappa** and **Krippendorff's Alpha** for measuring inter-rater reliability, and these metrics can be adapted to measure self-reliability as well. They should be preferred over metrics like accuracy, since as we saw in Table 3 that accuracy inflates agreement numbers because it does not take chance agreement into account.

Reduce Variance in LLM Outputs We saw in Section 5.1 that taking an aggregate of the results of multiple runs, like a simple majority vote, can improve agreement with human judgment. On the other hand, trying to eliminate variance entirely by turning off variance hurts performance.

Collect Data on Self-Reliability of Human Judges When collecting human evaluations on NLG tasks, it is important to also measure self-reliability data on those evaluations. This is because although we saw self-reliability is a problem in LLM judges, we did not explore how prevalent this problem is in human judges. As we consider human judgments the gold standard in NLG evaluation, the self-reliability of human judges would

serve as an upper bound of the self-reliability we should reasonably expect from LLM judges. It is also important to explore how much of an effect training or expertise has on self-reliability, since we saw in Section 5.2 that expert and crowdsourced workers assign very different ratings.

7 Conclusion

We present a comprehensive analysis of intra-rater reliability in LLM-as-a-judge frameworks, revealing key challenges for evaluation in this domain. Our experiments show that LLMs display low self-reliability when evaluating the same content across multiple runs, even with identical prompts and hyperparameters. This inconsistency persists across various tasks and metrics. Although newer, larger models like Qwen 3 are more consistent than models such as Llama 3.1, they still often fall short of standard reliability thresholds.

Our work carries some potential risks. We have measured the performance of LLM judges by their agreement with human judges, which can be noisy depending on the inter-annotator agreement of the human labels. LLM judgments could also potentially have biases for certain linguistic styles which could influence the scores presented in this paper. Despite these risks, our findings have important implications for LLM-as-a-judge research. Reporting single-run LLM judgments without consistency metrics can be misleading, and certain aspects of text quality, such as fluency, remain difficult to assess reliably for both humans and machines. Also, the pronounced disagreement between expert and crowdsourced judges highlights the need to clarify which human preferences LLMs should model.

While LLM-as-a-judge approaches offer scalability, they face significant reliability challenges. Addressing these issues is essential for developing evaluation frameworks that meaningfully capture distinctions in text quality.

Limitations

Recent research has looked at analyzing reasoning traces of LLM judges (Wang et al., 2023), whether these reasoning traces are useful and how their performance is affected by supervised fine-tuning or reinforcement learning (Chen et al., 2025). While we leveraged reasoning models like Deepseek R1 and Qwen 3 in our experiments, we have not explored any relationships between their reasoning traces and their self-reliability. And while prompting LLM judges on benchmarks like SummaC surpasses baselines in terms of agreement with human judgments, we have not looked at the effect of fine-tuning on further improving performance as that is beyond the scope of the paper. Beyond fine-tuning, there are other potential avenues worth exploring like investigating the impact of prompt structure on reliability and applying probing the model internals to study if a specific layer leads to the divergence in assigned scores in conflicting runs. Finally, though we identified that the subjectivity of summarization or multi-turn conversation evaluation potentially exacerbates the self-reliability issue of LLM judges, we have not quantified this subjectivity through comparison with self-reliability on other, more objective tasks. Nevertheless, this work highlights an important issue in evaluating generation tasks and discusses different experimental methods and data collection principles that would allow us to conduct more robust evaluations.

Finally, we limited our choice of models to those with open weights, as this gave us the freedom to run our experiments locally and with full access to the model weights. However, the latest proprietary models such as GPT-5 (OpenAI, 2025) and Claude-4 (Anthropic, 2025) have made great strides in various reasoning and agentic benchmarks. Along with the aforementioned future directions suggested, a comparison of the self-reliability in these models with the models in our study would give additional insights into the prevalence of this phenomenon.

Acknowledgments

This research used the Delta advanced computing and data resource which is supported by the National Science Foundation (award OAC 2005572) and the State of Illinois. Delta is a joint effort of the University of Illinois Urbana-Champaign and its National Center for Supercomputing Applications.

References

Gavin Abercrombie, Dirk Hovy, and Vinodkumar Prabhakaran. 2023a. Temporal and second language influence on intra-annotator agreement and stability in hate speech labelling. In *Proceedings of the 17th Linguistic Annotation Workshop (LAW-XVII)*, pages 96–103, Toronto, Canada. Association for Computational Linguistics.

Gavin Abercrombie, Verena Rieser, and Dirk Hovy. 2023b. Consistency is key: Disentangling label variation in natural language processing with intra-annotator agreement. *Preprint*, arXiv:2301.10684.

Anthropic. 2025. Introducing Claude 4: Claude Opus 4 and Claude Sonnet 4. https://www.anthropic.com/news/claude-4. Accessed: 2025-09-19.

Ron Artstein and Massimo Poesio. 2008. Survey article: Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Zahra Ashktorab, Michael Desmond, Qian Pan, James M. Johnson, Martin Santillan Cooper, Elizabeth M. Daly, Rahul Nair, Tejaswini Pedapati, Swapnaja Achintalwar, and Werner Geyer. 2024. Aligning human and llm judgments: Insights from evalassist on task-specific evaluations and ai-assisted assessment strategy preferences. *Preprint*, arXiv:2410.00873.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, André F. T. Martins, Philipp Mondorf, Vera Neplenbroek, Sandro Pezzelle, Barbara Plank, David Schlangen, Alessandro Suglia, Aditya K Surikuchi, Ece Takmaz, and Alberto Testoni. 2024. Llms instead of human judges? a large scale empirical study across 20 nlp evaluation tasks. *Preprint*, arXiv:2406.18403.

N. P. Bent, A. B. Rushton, C. C. Wright, E. J. Petherick, and M. E. Batt. 2014. Intrarater and interrater reliability of the anteromedial reach test in healthy participants. *Open Access Journal of Sports Medicine*, 5:1–10.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2021. Evaluation of text generation: A survey. *Preprint*, arXiv:2006.14799.

Nuo Chen, Zhiyuan Hu, Qingyun Zou, Jiaying Wu, Qian Wang, Bryan Hooi, and Bingsheng He. 2025. Judgelrm: Large reasoning models as a judge. *Preprint*, arXiv:2504.00050.

Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15607–15631, Toronto, Canada. Association for Computational Linguistics.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. 2024. Chatbot arena: An open platform for evaluating llms by human preference. *Preprint*, arXiv:2403.04132.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Yoav Cohen. 2017. Estimating the intra-rater reliability of essay raters. *Frontiers in Education*, 2.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Lance Eliot. 2025. Why doing chain-of-thought prompting in reasoning llms gums up the works.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Preprint*, arXiv:2007.12626.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. 2025. A survey on llm-as-a-judge. *Preprint*, arXiv:2411.15594.

Naomi Harvey. 2021. A simple guide to inter-rater, intra-rater and test-retest reliability for animal behaviour studies.

Dandan Huang, Leyang Cui, Sen Yang, Guangsheng Bao, Kun Wang, Jun Xie, and Yue Zhang. 2020. What have we achieved on text summarization? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 446–469, Online. Association for Computational Linguistics.

Hui Huang, Yingqi Qu, Xingyuan Bu, Hongli Zhou, Jing Liu, Muyun Yang, Bing Xu, and Tiejun Zhao. 2024. An empirical study of llm-as-a-judge for llm evaluation: Fine-tuned judge model is not a general substitute for gpt-4. *Preprint*, arXiv:2403.02839.

T. K. Koo and M. Y. Li. 2016. A guideline of selecting and reporting intraclass correlation coefficients for reliability research. *Journal of Chiropractic Medicine*, 15(2):155–163. Erratum in: J Chiropr Med. 2017 Dec;16(4):346. doi: 10.1016/j.jcm.2017.10.001.

Klaus Krippendorff. 2011. Computing krippendorff's alpha-reliability.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

A. Królikowska, P. Reichert, J. Karlsson, C. Mouton, R. Becker, and R. Prill. 2023. Improving the reliability of measurements in orthopaedics and sports medicine. *Knee Surgery, Sports Traumatology, Arthroscopy*, 31(12):5277–5285.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, Kai Shu, Lu Cheng, and Huan Liu. 2025. From generation to judgment: Opportunities and challenges of llm-as-a-judge. *Preprint*, arXiv:2411.16594.

Haitao Li, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu. 2024. Llms-as-judges: A comprehensive survey on llm-based evaluation methods. *Preprint*, arXiv:2412.05579.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

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*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

C. R. Mischiati, M. Comerford, E. Gosford, J. Swart, S. Ewings, N. Botha, M. Stokes, and S. L. Mottram. 2015. Intra and inter-rater reliability of screening for movement impairments: Movement control tests from the foundation matrix. *Journal of Sports Science and Medicine*, 14(2):427–440.

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. *Transactions of the Association for Computational Linguistics*, 12:933–949.

Harsha Nori, Naoto Usuyama, Nicholas King, S. McKinney, Xavier Fernandes, Sheng Zhang, and Eric Horvitz. 2024. From medprompt to o1: Exploration of run-time strategies for medical challenge problems and beyond.

OpenAI. 2025. Introducing GPT-5. `https://openai.com/index/introducing-gpt-5/`. Accessed: 2025-09-19.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia,

Pennsylvania, USA. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Shivalika Singh, Yiyang Nan, Alex Wang, Daniel D'Souza, Sayash Kapoor, Ahmet Üstün, Sanmi Koyejo, Yuntian Deng, Shayne Longpre, Noah A. Smith, Beyza Ermis, Marzieh Fadaee, and Sara Hooker. 2025. The leaderboard illusion. *Preprint*, arXiv:2504.20879.

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. *Preprint*, arXiv:2410.12784.

Aman Singh Thakur, Kartik Choudhary, Venkat Srinik Ramayapally, Sankaran Vaidyanathan, and Dieuwke Hupkes. 2025a. Judging the judges: Evaluating alignment and vulnerabilities in llms-as-judges. *Preprint*, arXiv:2406.12624.

Aman Singh Thakur, Kartik Choudhary, Venkat Srinik Ramayapally, Sankaran Vaidyanathan, and Dieuwke Hupkes. 2025b. Judging the judges: Evaluating alignment and vulnerabilities in LLMs-as-judges. In *Proceedings of the Fourth Workshop on Generation, Evaluation and Metrics (GEM²)*, pages 404–430, Vienna, Austria and virtual meeting. Association for Computational Linguistics.

Craig Thomson, Ehud Reiter, and Anya Belz. 2024. Common flaws in running human evaluation experiments in nlp. *Computational Linguistics*, 50(2):795–805.

Yu-Min Tseng, Wei-Lin Chen, Chung-Chi Chen, and Hsin-Hsi Chen. 2024. Are expert-level language models expert-level annotators? *Preprint*, arXiv:2410.03254.

Pat Verga, Sebastian Hofstatter, Sophia Althammer, Yixuan Su, Aleksandra Piktus, Arkady Arkhangorodsky, Minjie Xu, Naomi White, and Patrick Lewis. 2024. Replacing judges with juries: Evaluating llm generations with a panel of diverse models. *Preprint*, arXiv:2404.18796.

Nico Wagner, Michael Desmond, Rahul Nair, Zahra Ashktorab, Elizabeth M. Daly, Qian Pan, Martín Santillán Cooper, James M. Johnson, and Werner Geyer. 2024. Black-box uncertainty quantification method for llm-as-a-judge. *Preprint*, arXiv:2410.11594.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

M. Wiethölter, D. Akgün, F. Plachel, M. Minkus, D. Karczewski, K. Braun, K. Thiele, L. Becker, U. Stöckle, and P. Moroder. 2023. Inter-observer and intra-observer reliability assessment of the established classification systems for periprosthetic shoulder fractures. *Journal of Clinical Medicine*, 12(9):3168.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Ziang Xiao, Susu Zhang, Vivian Lai, and Q. Vera Liao. 2023. Evaluating evaluation metrics: A framework for analyzing NLG evaluation metrics using measurement theory. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10967–10982, Singapore. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

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. *Preprint*, arXiv:2306.05685.

A More Details About Experimental Setup

For all three benchmarks and all three LLMs in our experiments, we used a 4xA100 GPU server and the transformers library (Wolf et al., 2020) to run our experiments. Unless specified otherwise, we used the recommended defaults for each model. This means that for Llama 3.1 and DeepSeek-R1, we used a temperature of 0.6 and top_p of 0.9. For

Qwen 3, we used a temperature of 0.6 and top_p of 0.95.

For the **SummaC** benchmark, we used the following prompt to get a binary label representing whether the article is consistent or inconsistent. Figure 8 shows the prompt used for evaluating items in this benchmark.

Task: Analyze the summary for factual inconsistencies against the source document. Inconsistencies can be due to:

1. **Hallucinations**: Information added not in the source.
2. **Contradictions**: Statements opposing source content.
3. **Entity Errors**: Incorrect names/roles/locations.
4. **Omissions**: Key points missing from the summary.
5. **Temporal Errors**: Wrong sequence/timeframe of events.

Output: A single number **0** for consistent summary and **1** for inconsistent summary.
Document: {{Full source text}}
Summary: {{Generated Summary}}

Figure 8: Prompt used for each run in SummaC benchmark.

For **SummEval**, there are four different metrics, coherence, consistency, fluency and relevance. For each run, we prompted the LLM judge four times independently, once for each metric. The prompts are in Figure 9.

For MTBench, we used a single prompt that asked an LLM judge to choose between two generations, or assign the label *tie* if it was unable to choose a clear favorite. Items in this datasets are divided into two types, *math* and *general*, with the difference being that conversations labeled *math* also contain reference answers to the user query. Figure 10 shows the prompt used for items labeled *math*, while Figure 11 shows the prompt used for items labeled *general*.

B Agreement Metric: Krippendorff's Alpha

Krippendorff's alpha (α) is a chance-corrected reliability coefficient used to quantify agreement among two or more coders (observers, raters) assigning values to a set of units. In other words, it measures how consistently multiple annotators code the same items. Developed by Klaus Krippendorff in the context of content analysis, α generalizes many classical agreement statistics (such as Cohen's (Cohen, 1960) or Fleiss' (Fleiss, 1971) Kappa) and applies to any number of coders, any number of categories or scale values, and any measurement level (nominal, ordinal, interval, ratio, etc.)

Krippendorff's alpha accommodates any number of raters (two or more) and any measurement level, from nominal categories up through interval (or ratio) scales. It can handle missing data (by simply omitting those cases when counting pairwise judgments), and it yields comparable reliability coefficients even for unequal sample sizes or many categories. This flexibility allows it to remain valid for more than two raters or for ordered categories where partial agreements matter. Overall, Krippendorff's alpha is valued for its broad applicability and its principled treatment of chance agreement, making it suitable for diverse rating and annotation tasks.

B.1 Mathematical Notation

Mathematically, Krippendorff's Alpha follows the general form:

$$\alpha = 1 - \frac{D_o}{D_e} \tag{1}$$

Here, $D_o = \sum \delta(v_i, v_j)$ sums the pairwise distances between raters' values v_i, v_j on each item.

The denominator D_e is computed by summing δ over all possible pairs of values (weighted by the frequency of each value) as if the assignments were randomly permuted among raters. In other words, D_e reflects the average disagreement expected purely by chance given the observed distribution of categories. The expected disagreement D_e is what separates Krippendorff's Alpha from other metrics like correlation for percentage agreement, since it accounts for chance agreement to prevent inflation of agreement values. To understand how D_e is calculated, let us take the following example where we have

1. V which is the set of all possible values that coders can assign (e.g., $V = \{1, 2, 3, 4, 5\}$ for a 5-point scale).

2. n_v which is the number of times value $v \in V$ was assigned, across all units and coders.

Instructions: You will be given one summary written for a news article.
Your task is to rate the summary on one metric.
Evaluation Criteria: Coherence (1–5) – how well the summary is structured and logically organized.
Evaluation Steps:
1. Read article and identify key points.
2. Check if summary presents them clearly and logically.
3. Score 1–5.

Example:
News Article:
{{Source Text}}
Summary:
{{Summary}}
Evaluation Form (scores ONLY):
Coherence:

(a) Coherence

Instructions: You will be given one summary written for a news article.
Your task is to rate the summary on one metric.
Evaluation Criteria: Consistency (1–5) – the summary should not contradict the source; penalize hallucinated facts.
Evaluation Steps:
1. Read article and summary.
2. Identify any factual errors.
3. Score 1–5.

Example:
News Article:
{{Source Text}}
Summary:
{{Summary}}
Evaluation Form (scores ONLY):
Consistency:

(b) Consistency

Instructions: You will be given one summary written for a news article.
Your task is to rate the summary on one metric.
Evaluation Criteria: Fluency (1–5) – grammar, spelling, punctuation, word choice, and sentence structure.
Evaluation Steps:
1. Read the summary.
2. Identify language issues affecting readability.
3. Score 1–5.

Example:
Summary:
{{Summary}}
Evaluation Form (scores ONLY):
Fluency:

(c) Fluency

Instructions: You will be given one summary written for a news article.
Your task is to rate the summary on one metric.
Evaluation Criteria: Relevance (1–5) – includes only important information from the source; penalize redundancy.
Evaluation Steps:
1. Read summary and article.
2. Assess coverage of key points.
3. Score 1–5.

Example:
News Article:
{{Source Text}}
Summary:
{{Summary}}
Evaluation Form (scores ONLY):
Relevance:

(d) Relevance

Figure 9: Prompts For Evaluating Generated Summaries From SummEval Using Four Metrics

Please act as an impartial judge and evaluate the quality of the responses provided by two AI assistants to the user questions. Your evaluation should consider correctness and helpfulness. You will be given reference answers, the assistant A's answers, the assistant B's answers. Your job is to determine which assistant provides correct and helpful answers to the second user question. Begin your evaluation by comparing both assistants' answers with the reference answers. Identify and correct any mistakes. Avoid any position biases and ensure that the order in which the responses were presented does not influence your decision. Do not allow the length of the responses to influence your evaluation. Do not favor certain names of the assistants. Be as objective as possible. After providing your explanation, output your final verdict by strictly following this format: "[[A]]" if assistant A is better, "[[B]]" if assistant B is better, and "[[C]]" for a tie.

<|The Start of Reference Answer|>

User:
{{question_1}}
Reference answer:
{{ref_answer_1}}
User:
{{question_2}}
Reference answer:
{{ref_answer_2}}

<|The End of Reference Answer|>

<|The Start of Assistant A's Conversation with User|>

User:
{{question_1}}
Assistant A:
{{answer_a_1}}
User:
{{question_2}}
Assistant A:
{{answer_a_2}}

<|The End of Assistant A's Conversation with User|>

<|The Start of Assistant B's Conversation with User|>

User:
{{question_1}}
Assistant B:
{{answer_b_1}}
User:
{{question_2}}
Assistant B:
{{answer_b_2}}

<|The End of Assistant B's Conversation with User|>

Figure 10: Prompt for Evaluating Two Generated Responses to Queries Labeled *math* in MTBench Dataset

Please act as an impartial judge and evaluate the quality of the responses provided by two AI assistants to the user questions. You should choose the assistant that follows the user's instructions and answers the user's questions better. Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of their responses. You should focus on who provides a better answer to the second user question. Begin your evaluation by comparing the responses of the two assistants and provide a short explanation. Avoid any position biases and ensure that the order in which the responses were presented does not influence your decision. Do not allow the length of the responses to influence your evaluation. Do not favor certain names of the assistants. Be as objective as possible. After providing your explanation, output your final verdict by strictly following this format: "[[A]]" if assistant A is better, "[[B]]" if assistant B is better, and "[[C]]" for a tie.

<|The Start of Assistant A's Conversation with User|>

User:
{{question_1}}
Assistant A:
{{answer_a_1}}
User:
{{question_2}}
Assistant A:
{{answer_a_2}}

<|The End of Assistant A's Conversation with User|>

<|The Start of Assistant B's Conversation with User|>

User:
{{question_1}}
Assistant B:
{{answer_b_1}}
User:
{{question_2}}
Assistant B:
{{answer_b_2}}

<|The End of Assistant B's Conversation with User|>

Figure 11: Prompt for Evaluating Two Generated Responses to Queries Labeled *general* in MTBench Dataset

3. $N = \sum_{v \in V} n_v$ which is the total number of value assignments.

From this, we can compute the relative frequencies (probabilities):

$$p_v = \frac{n_v}{N} \tag{2}$$

Now we compute the expected disagreement by summing over all unordered pairs of values $(v, v') \in V \times V$, weighted by how frequently those value pairs would co-occur by chance if coder assignments were independent.

We define a weighting term:

- If $v \neq v'$, the probability of randomly drawing the pair (v, v') is $2 \cdot p_v \cdot p_{v'}$. The factor of 2 ensures we're counting both (v, v') and (v', v) as one unordered pair.
- If $v = v'$, the probability of drawing (v, v) is p_v^2

Using a distance function $\delta(v, v')$, the expected disagreement is:

$$D_e = \sum_{v \in V} \sum_{v' \in V} p_v p_{v'} \, \delta(v, v') \tag{3}$$

Conceptually, this represents the expected value of the disagreement between two randomly selected values from the overall distribution.

When observers agree perfectly, $D_o = 0$ and $\alpha = 1$, indicating perfect reliability. When observers agree only by chance, $D_o = D_e$ and $\alpha = 0$, indicating absence of reliability.

B.2 Distance Functions

A crucial aspect of Krippendorff's alpha is the distance function $\delta(v, v')$, which quantifies how much two coded values disagree. The choice of δ reflects the measurement level of the data and fundamentally affects the computation of disagreement. In general δ must satisfy $\delta(v, v') = 0$ when two raters agree and $\delta(v, v') > 0$ otherwise. Different standard choices of δ are used for nominal, ordinal, and interval data. Each distance function changes how disagreement is accumulated in D_o and D_e.

B.2.1 Nominal Distance

For nominal (categorical) data with no inherent order, the standard distance function is the discrete (binary) metric. In this case, any two different categories are simply considered maximally different. Formally, one sets

$$\delta_{\text{nominal}}(v, v') = \begin{cases} 0, & v = v' \\ 1, & v \neq v' \end{cases} \tag{4}$$

Therefore, exact matches incur zero distance and any mismatch contributes a distance of 1. Using this δ means the observed disagreement D_o is effectively the count (or weighted count) of all pairwise coding disagreements among raters.

To calculate D_o using nominal distance,

$$D_o = \sum_{\text{pairs}} \delta_{\text{nominal}}(v_i, v_j) \tag{5}$$

So D_o is essentially the count of mismatches.

Nominal distance is appropriate when values are simply labels or categories (e.g. color names, types of object, or coding categories like "yes/no", "red/green/blue" etc.). Here there is no notion of "how different" two distinct categories are, only that they are different. In effect, α with nominal δ reduces to a chance-corrected proportion of exact-agreement (similar in spirit to kappa). All disagreements are weighted equally, so a mild coding error ("cat" vs "dog") counts the same as a gross one ("cat" vs "car") in D_o. This is the metric we used in the SummaC benchmark, since the items there have binary labels.

B.2.2 Ordinal Distance

For ordinal data (ranked categories, e.g. survey responses "low/medium/high" or Likert scales), the distance function must respect the ordering of values. A common choice is to use the squared difference in ranks (or a formula based on cumulative frequencies of ranks). Krippendorff's original formulation defines the ordinal distance as

$$\delta_{ordinal}(v, v') = \left(\sum_{g=\min(v,v')}^{\max(v,v')} n_g - \frac{n_v + n_{v'}}{2} \right)^2 \tag{6}$$

Here, n_v is the frequency of category v. In simpler terms, this usually reduces to the squared difference in rank position between vv and v'v', possibly standardized by category frequencies. The key effect is that $\delta_{ordinal}(v, v')$ increases as the categories are farther apart in rank.

For ordinal δ, if N units are coded and unit j has m_j coders giving values $v_{1j}, \ldots, v_{m_j j}$, then the total number of coder-pairs is

$$n = \sum_{j=1}^{N} \binom{m_j}{2} \tag{7}$$

Using the ordinal distance δ_{ord}, Krippendorff's D_o can be written as

$$D_o = \frac{1}{n} \sum_{j=1}^{N} \sum_{i<i'} \delta_{ord}(v_{ij}, v_{i'j}) \tag{8}$$

Here each inner sum runs over all $\binom{m_j}{2}$ distinct pairs of coders (i, i') within unit j. Equivalently (and as shown by Krippendorff), this is the weighted average of within-unit disagreements. In other words, D_o is the mean of all squared ordinal distances between pairs of ratings of the same unit.

Each category has a rank (e.g. $1, 2, 3, \ldots$). The standard ordinal distance (squared) between two categories c and k is defined by how far apart their ranks lie in the observed data distribution. Specifically

$$\delta_{ord}(c, k) = \left(\sum_{g=\min(c,k)}^{\max(c,k)} n_g - \frac{n_c + n_k}{2} \right)^2 \tag{9}$$

where n_g is the frequency of category g in the pooled data. Intuitively, $\delta_{\text{ord}}(c, k)$ counts the total number of cases between c and k (plus half-counts of the endpoints), then squares that gap. Plugging this into D_o means each pair of ratings contributes the square of the rank-gap between their categories.

Ordinal distance is appropriate when categories are ordered but not equally spaced (e.g. ratings like "good, better, best" or "strongly agree, agree, neutral, disagree, strongly disagree"). Using this distance in α means that two raters who give adjacent ranks (e.g. 2 vs. 3 on a five-point scale) incur less disagreement than two raters who give opposite ends (e.g. 1 vs. 5). In formula terms, D_o will sum the squared rank differences for each pair. Thus α will penalize large rank disagreements more heavily. In practice, this often yields a larger D_o (hence smaller α) than the nominal distance would, because it encodes more information about how coders differ, not just that they differ. (One can also standardize ordinal distances to lie between 0 and 1, but the relative weighting is what matters for α.) This is the distance we used in the SummEval and MTBench benchmarks, since the former has labels on a Likert Scale, and the latter has three labels, $model_a$, $model_b$ and tie, allowing us to distinguish between disagreements where get ($model_a$, tie) versus ($model_a$, $model_b$).

B.2.3 Interval Distance

For interval data (quantitative values on a scale with equal intervals), the usual choice is the squared-difference distance. That is, one sets

$$\delta_{interval}(v, v') = (v - v')^2 \tag{10}$$

This simply treats the numerical values as points on the real line and measures squared distance between them.

To calculate D_o from interval distance,

$$D_o = \sum_{\text{pairs}} (v_i - v_j)^2 \tag{11}$$

So D_o is the sum of squared deviations across all coder pairs.

Interval distance is used when the data are measured on an interval scale (e.g. temperature in Celsius, or any numeric rating where differences are meaningful). For example, if raters assign values 3.0 and 4.5 to an item, $\delta = (3.0 - 4.5)^2 = 2.25$. In α's computation, each such pairwise difference contributes to the total disagreement. The squared form makes α analogous to a variance-based agreement measure: larger numerical discrepancies inflate D_o. In effect, with interval δ, Krippendorff's alpha becomes sensitive to the magnitude of disagreements. Two coders who differ by 1 unit vs. two coders who differ by 5 units will have drastically different contributions to D_o. This is appropriate when numerical differences carry substantive meaning. We do not use this metric in this paper, since none of the datasets in our studies contain numerical labels, however, we are still including this distance function in our explanation for the sake of completeness.

C More Details on SummaC Benchmark

C.1 Dataset Statistics of SummaC Benchmark

Table 4 shows the number of test examples in each of the six datasets in the SummaC benchmark.

Dataset	Test Examples
CoGenSumm (Falke et al., 2019)	400
XSumFaith (Maynez et al., 2020)	1250
Polytope (Huang et al., 2020)	634
FactCC (Kryscinski et al., 2020)	503
SummEval (Fabbri et al., 2021)	850
FRANK (Pagnoni et al., 2021)	1575

Table 4: Number of Test Examples in Each Dataset in the SummaC Benchmark

C.2 Impact of CoT and Few-shot Prompting on Intra-Rater Reliability and Accuracy on SummaC

Two breakthrough advances in the use of language models are few-shot prompting (Brown et al., 2020) and chain-of-thought (CoT) prompting (Wei et al., 2022). Few-shot prompting refers to providing the model with a few labeled examples demonstrating how to perform a desired task with higher accuracy. On the other hand, CoT prompting enables language models to generate intermediate reasoning steps that lead to the desired answer. They represent different facets of in-context learning, and can be combined or used independently.

Setting	Llama 3.1	Deepseek-R1	Qwen 3
Default	0.3263	0.6278	0.7883
w/ Few-shot	0.3245	0.6166	0.7804
w/ CoT	0.3219	0.6132	0.7796
w/ Both	0.3206	0.6134	0.7801

Table 5: Self-Reliability of LLM judges on SummaC for the default setting, as well as when few-shot and chain-of-thought prompting are used together and separately.

Setting	Llama 3.1	Deepseek-R1	Qwen 3
Default	61.4	72.3	80.6
w/ Few-shot	62.9	72.8	80.6
w/ CoT	60.8	71.6	80.4
w/ Both	60.1	72.1	80.3

Table 6: Balanced Accuracy (majority-vote) between the LLM and human judgments for the default setting, as well as when few-shot and chain-of-thought prompting are used together and separately.

In our next set of experiments, we investigate whether leveraging these techniques addresses self-reliability. In the few-shot setting, we add 5 positive and 5 negative examples in the prompt. In the chain-of-thought setting, we prompt the model to think step by step before answering. Under the Both setting, we employ both few-shot and chain-of-thought prompting.

Table 5 shows the self-reliability of LLM judges measured by Krippendorff's Alpha over 3 runs. We see that few-shot prompting does not appreciably change the self-reliability of the models, and leads to a slight decrease in consistency. This change is not significant and shows that one cannot easily address the self-reliability issue with different prompting strategies.

Table 6 shows the accuracy of LLM judges against human judgments. While few-shot prompting does lead to a small improvement in Llama 3.1 and Deepseek-R1, there is no significant jump in performance for either prompting strategy. This could be because, as previously observed, few-shot prompting does not always work well with reasoning models like Deepseek-R1, Qwen 3 and

o1 (DeepSeek-AI et al., 2025; Nori et al., 2024), while chain-of-thought prompting does not necessarily improve the accuracy of a reasoning model since it already performs chain-of-thought implicitly (Eliot, 2025). Interestingly, Qwen 3 shows the least change under these new settings, suggesting newer reasoning models are fairly robust to different prompts.

D Additional Analysis of Agreement Between Judges in SummEval

D.1 Distributions of Scores in SummEval From Different Populations of Raters

In Figure 12, we see the distributions of scores assigned by different categories of raters across all four metrics on the SummEval benchmark. We see that for Turkers the distributions do not change appreciably across metrics, indicating that human judges without proper expertise do a poor job of differentiating between different metrics. While the other four categories of judges do discern between metrics, they all have very different distributions of scores for a specific metric. Even though Deepseek R1 and Qwen 3 showed much higher agreement with experts compared to LLama 3.1 and Turkers, we see that their distributions are very different. This implies that each category of judge follows their own internal definition of a given metric, which puts a strict cap on how high their agreement can be.

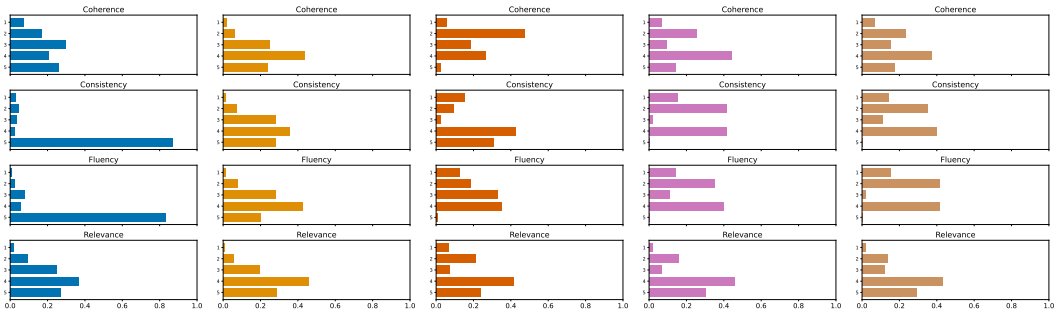
E Additional Details on MT-Bench Benchmark

E.1 Breakdown of Examples in MT-Bench

2	3	4	5
599	132	24	6

Table 7: Number of examples with 2, 3, 4, or 5 judgments from human annotators in MT-bench.

Table 7 shows how many examples had 2, 3, 4, or 5 human judgments in the MT-Bench dataset. These are the examples we used in our experiments since we filtered out examples that had only one human judgment.



(a) Experts (b) Turkers (c) Llama 3.1 (d) Deepseek R1 (d) Qwen 3

Figure 12: Distributions of scores assigned by different raters across all metrics in the SummEval dataset.