Think-J: Learning to Think for Generative LLM-as-a-Judge

Hui Huang[1]*, Yancheng He[1]*, Hongli Zhou[1]*, Rui Zhang[1], Wei Liu[1], Weixun Wang[1], Jiaheng Liu[2]†, Wenbo Su[1]

[1]Alibaba Group, Hangzhou, China
[2]School of Intelligence Science and Technology, Nanjing University, Suzhou, China
hh456524@taobao.com, liujiaheng@nju.edu.cn

Abstract

LLM-as-a-Judge refers to the automatic modeling of preferences for responses generated by Large Language Models (LLMs), which is of significant importance for both LLM evaluation and reward modeling. Although generative LLMs have made substantial progress in various tasks, their performance as LLM-Judge still falls short of expectations. In this work, we propose Think-J, which improves generative LLM-as-a-Judge by learning how to think. We first utilized a small amount of curated data to develop the model with initial judgment thinking capabilities. Subsequently, we optimize the judgment thinking traces based on reinforcement learning (RL). We propose two methods for judgment thinking optimization, based on offline and online RL, respectively. The offline method requires training a critic model to construct positive and negative examples for learning. The online method defines rule-based reward as feedback for optimization. Experimental results showed that our approach can significantly enhance the evaluation capability of generative LLM-Judge, surpassing both generative and classifier-based LLM-Judge without requiring extra human annotations.

Code — https://github.com/huihuichyan/think-j

1 Introduction

As the capabilities of generative LLMs continue to advance, accurately evaluating the response quality has emerged as a crucial challenge(Huang et al. 2025). This is not only vital for more efficient model development and comparison but also essential in the context of Reinforcement Learning from Human Feedback (RLHF), which relies on precise preference modeling as guidance (Wang et al. 2024a; He et al. 2025; Liu et al. 2025c). However, traditional evaluation methods for generative models, such as BLEU (Papineni et al. 2002), are based on predefined reference answers, which are often unavailable in open-ended scenarios.

Some studies have proposed LLM-as-a-Judge (Zheng et al. 2023), which leverages the generative capabilities of LLMs for evaluating response quality. These work either directly leverage proprietary LLMs or fine-tune a smaller judge based on preference data (Gu et al. 2024). However, the accuracy of their judgments remains unsatisfactory as revealed by recent benchmarks (Lambert et al. 2024). Other research has suggested fine-tuning a classifier based on preference data (Liu et al. 2024a). While this method can achieve higher judgment accuracy, it lacks interpretability due to its scalar output, and the performance is highly dependent on the data quality (Wang et al. 2024b).

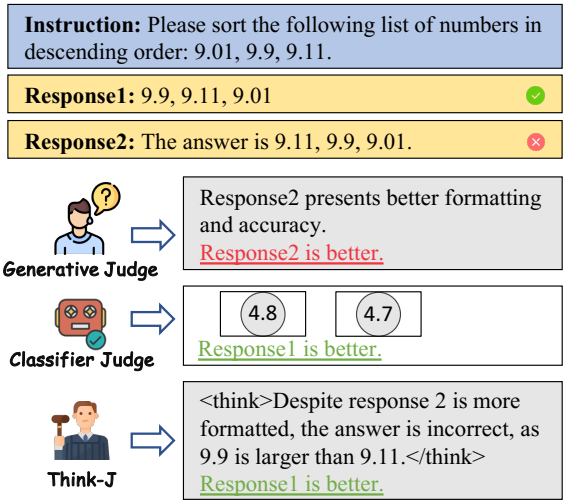

Figure 1: Comparison of different judge models. Our proposed Think-J takes into account both accuracy and interpretability based on thinking optimization.

Inspired by recent reasoning models such as o1 (Jaech et al. 2024) and Deepseek-R1 (Guo et al. 2025), in this work, we propose Thinking-enhanced Generative Judge (Think-J), as shown in Figure 1. Think-J aims to train a better generative judge by optimizing the model's judgment thinking capabilities. Specifically, Think-J consists of two steps:

1) Judgment Thinking Initialization. We carefully curated 707 samples from preference data, considering various aspects such as accuracy, difficulty, and diversity. After that, thinking trace is annotated by proprietary models to initialize the thinking capabilities of the judge.

2) Judgment Thinking Optimization. Due to the lack of high-quality critique annotation in preference datasets, we opt to optimize judgment thinking ability based on rein-

* Equal contribution.
† Corresponding Author.

forcement learning (RL). Specifically, we adopted two methods: a) Critic-guided Offline Learning, leveraging an additional critic model to generate corresponding thinking traces based on provided judgment results, thus constructing positive and negative examples for offline RL. b) Rule-based Online Learning, defining rule-based rewards based on the correctness of judgment results, thus optimizing the thinking trace by online RL (Shao et al. 2024).

We conducted experiments on three open-source models, and results showed that our proposed Think-J significantly outperformed existing LLM-judges with only limited training data. We also verify the effectiveness of our method compared with generative and classifier-based preference modeling methods. Our contributions are as follows:

1. We propose to stimulate the judgment thinking ability of generative models with carefully curated data.
2. We propose to optimize the judgment thinking ability of generative models with reinforcement learning.
3. Our proposed Think-J significantly outperforms previous generative and classifier-based LLM-as-judges.

2 Background

After the emergence of LLMs, numerous efforts have been made to design a more effective method for LLM evaluation (Chang et al. 2023). One of the most scalable and effective methods is LLM-as-a-Judge (Li et al. 2023b; Zheng et al. 2023), namely utilizing proprietary LLMs, especially GPT4 (Achiam et al. 2023), to evaluate the LLM's response. For example, AlpacaEval (Li et al. 2023b) used the win rate compared with baseline response determined by GPT-4 as the evaluation result. MT-Bench (Zheng et al. 2023) automatically scored the model's answers using GPT-4 as the results. The GPT-4-based evaluator is proven to presents comparable or even better consistency compared with human.

However, relying on external API for evaluation may introduce consideration about privacy leakage, and the opacity of API models also challenges the evaluation reproducibility. Therefore, follow-up works suggest fine-tuning language models locally for evaluations, including JudgeLM (Zhu, Wang, and Wang 2023), Auto-J (Li et al. 2023a), Prometheus (Kim et al. 2023), Prometheus-2 (Zhu et al. 2023), OffsetBias (Park et al. 2024), etc. These work typically construct preference data with judgment annotations and then finetune open-sourced LLMs to generate the judgment. Despite these fine-tuned judge models all achieve comparable accuracy with proprietary models, the evaluation is mostly conducted on the in-domain testsets, and these works are verified with a low scalability on more general benchmarks (Huang et al. 2024).

Another group of work fine-tunes a classifier on preference data based on the Bradley-Terry model, which is more commonly used on reward modeling. This approach is simple yet effective, as demonstrated on RewardBench (Lambert et al. 2024) where most top-performing models are trained in a classification style (Liu et al. 2024a). However, this method does not fully leverage the generative capabilities of LLMs, and is unable to provide rationales for its judgments, which is crucial for scalable evaluation. While recent work has begun to leverage the generative abilities of LLMs to combine critiques for scalar reward prediction (Ke et al. 2024; Ye et al. 2024), these critiques are often distilled from stronger proprietary models and hardly influence the final prediction (Liu et al. 2025d). Effectively integrating the generative abilities of LLMs into evaluation remains an open challenge (Chen et al. 2025a; Whitehouse et al. 2025; Chen et al. 2025b; Wang et al. 2025a).

Skywork-Reward-Preference-v0.2	Data Num
- initially selected	10K
- filtered by difficulty	1496
- filtered by diversity	870
- filtered by accuracy	707

Table 1: Data statistics during constructing LIMJ707.

3 Methodology

3.1 Judgment Thinking Initialization

Recent studies have shown that LLMs inherently possess long chain-of-thought (CoT) reasoning capabilities, which can be activated with a small amount of data (Muennighoff et al. 2025; Ye et al. 2025; Liu et al. 2025a). In this work, we also curate high-quality preference data, **LIMJ707**, to initialize the thinking capability of the judge model. Specifically, LIMJ707 is selected based on three principles:

- **Accuracy**: The judgment (preference) annotation should be correct. We leverage the high-quality preference data Skywork-Preference-v0.2[1], which has been carefully validated to ensure accurate annotation.
- **Difficulty**: The sample should be sufficiently challenging. We apply the judge models to perform judgment for the sample three times, and select those samples where at least one judgment is failed, as these samples are likely more difficult and reflect the insufficiency of the judge.
- **Diversity**: The instruction should encompass various types to enhance judgment thinking capabilities in different aspects. We represent the instructions with an embedding model and then merge duplicate samples[2].

The data statistics during processing are shown in Table 1. Based on LIMJ707, we construct judgment thinking trace by Deepseek-R1. After that, the annotated samples are used to initialize the model with judgment thinking capability by Supervised Fine-tuning (Ouyang et al. 2022)[3].

3.2 Judgment Thinking Optimization

To further enhance the alignment between the judge and human preference, the initialized thinking trace should be further optimized on preference data. However, preference data

[1]huggingface.co/datasets/Skywork/Skywork-Reward-Preference-80K-v0.2

[2]For more details please refer to Appendix B.1.

[3]Due to the overly long thinking trace generated by Deepseek-R1, we performed trace-clipping to reduce training overhead and improve efficiency. Please refer to Appendix B.1 for more details.

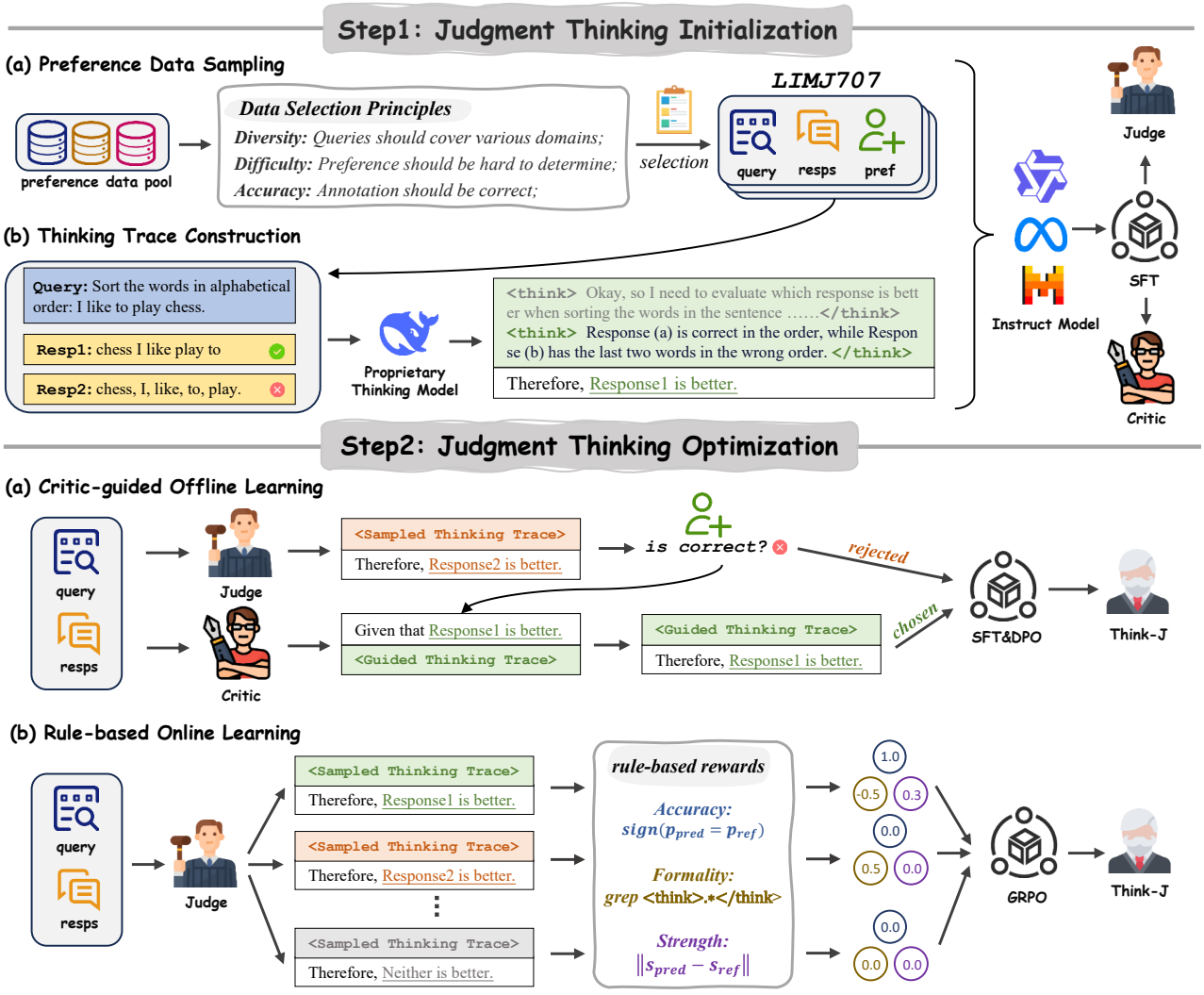

Figure 2: The illustration of our proposed framework. We begin by constructing high-quality judgment thinking traces using curated principles and proprietary thinking models. Based on this data, we initialize a judge model and a critic model, both equipped with judgment thinking capability. After that, we optimize the capability of the judge model through two methods: Critic-guided Offline Learning and Rule-based Online Learning, resulting in Think-J.

typically only includes binary labels without thinking trace annotations. Therefore, we propose judgment thinking optimization based on reinforcement learning (RL). Specifically, we propose two methods based on offline and online learning respectively, as shown in Figure 2.

Critic-guided Offline Learning Offline RL methods represented by Direct Preference Optimization (DPO) (Rafailov et al. 2023) has been widely applied to LLM pipelines due to their efficiency and simplicity (Grattafiori et al. 2024). In this work, we also aim to optimize the judgment thinking ability based on offline learning.

Due to the lack of golden thinking annotation in preference dataset[4], we propose to train an additional critic model[5], to help to construct training samples. Both the critic and the judge models are trained on the same data (i.e., LIMJ707), with the following distinctions:

- **Judge Model:** Given instruction-responses, it generates the thinking trace and judgment result.
- **Critic Model:** Given instruction-responses and judgment result, it generates the thinking trace.

Based on the two models, we can perform thinking optimization with the following steps:

1. First, leverage the judge to evaluate the input to generate the thinking traces and results.
2. If the result is correct, use the critic to generate an incorrect trace as the negative sample. Conversely, if the result

[4]The rationale for fine-tuning an additional critic model over sampling to generate preference data is presented in Appendix A.3.

[5]Notice the critic model here differs from the critic model in traditional RL which is used for advantage estimation.

is incorrect, use the critic to generate a correct trace as the positive sample.

3. Based on the positive and negative samples, optimize the judgment thinking ability with offline learning objective.
4. These steps can be iterated to continuously enhance the judgment thinking capability.

We adopt a combination of SFT and DPO as our training objective in this step:

$$\mathcal{L}_{\text{offline}}(\pi_\theta; \mathcal{D}) = - E_{x\sim\mathcal{D},(y_w,y_l)\sim\pi_\theta(y|x)} \left[\log \pi_\theta(y_w \mid x) + \log \sigma \left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{ref}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{ref}(y_l \mid x)}\right)\right] \quad (1)$$

where π_θ and π_{ref} represent the policy model and the reference model, and y_w and y_l denotes the positive and negative judgment traces, respectively.

With the help of critic model, samples with thinking annotation are constructed based on the correctness of judgment result. Therefore, the judge model will be enhanced to generate more accurate thinking for better judgment.

Rule-based Online Learning The recent success of R1-style methods have demonstrated the effectiveness of online RL using discrete, rule-based rewards (Shao et al. 2024). In this work, we also apply online rule-based RL approach to optimize the judgment thinking capability. More specifically, we mainly utilize the GRPO algorithm, with the optimization objective as follows:

$$J_{\text{online}}(\pi_\theta; \mathcal{D}) = E_{x\sim\mathcal{D},\{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(y|x)} \left[\frac{1}{G}\sum_{i=1}^{G} \min\left(\frac{\pi_\theta(y_i|x)}{\pi_{\theta_{\text{old}}}(y_i|x)} A_i, \text{clip}\left(\frac{\pi_\theta(y_i|x)}{\pi_{\theta_{\text{old}}}(y_i|x)}, 1-\epsilon, 1+\epsilon\right) A_i\right) - \beta D_{\text{KL}}(\pi_\theta||\pi_{\text{ref}})\right] \quad (2)$$

where G is group size, and A_i is advantage. The reward function is designed as follows[6]:

$$r_{\text{accuracy}} = \begin{cases} 1, & \text{if judgement} = \text{label} \\ 0, & \text{if judgement} \neq \text{label} \end{cases} \quad (3)$$

$$r_{\text{format}} = \begin{cases} 0, & \text{if format is right} \\ -0.5, & \text{if format is wrong} \end{cases} \quad (4)$$

$$r_{\text{strength}} = ||\text{s}_{\text{pred}} - \text{s}_{\text{golden}}||, s \in \{1, 2, 3\} \quad (5)$$

$$r_{\text{final}} = \alpha \cdot r_{\text{accuracy}} + \beta \cdot r_{\text{format}} + \gamma \cdot r_{\text{strength}} \quad (6)$$

where detailed weights for different rewards are presented in Section 5.2. Notice we incorporate a reward for assessing preference strength, which is defined as the degree to which the judge favors one response over another[7]. The prompt template of judgment is also adjusted as:

```
<think> {thinking trace} </think>
Therefore, Response (a) is better, and the preference strength is [[2]].
```

Preference strength helps to perceive the relative quality of response pairs, without uniformly providing the same reward for different pairs despite the quality gap. We employ a comparatively simple scale of reward scores of reward, as during actual training, we observed that the model tends to manipulate scores towards extreme values[8].

While absolute score annotation for a given response is challenging, annotating relative preference strength is more readily accessible (Wang et al. 2024d, 2025b). Moreover, if absolute score annotation is absent in the preference data, we can firstly fine-tune a BT-classifier based on the data, then leverage the classifier to assess the relative strength between chosen and rejected responses (Wang et al. 2024b).

4 Experiments

4.1 Set-up

We mainly conducted experiments on two popular open-sourced models with their instruction version: Qwen-2.5 (Qwen et al. 2025) and Llama-3 (Grattafiori et al. 2024).

We primarily conducted our optimization on two datasets: HelpSteer2-Pref[9] and HH-RLHF[10]. For a fair comparison, LIMJ707 was mixed into all training sets. While larger preference datasets are available, we excluded them from our training because our primary objective was to explore the most effective method for training LLM judges.

We mainly compare Think-J with the following generative LLM-judge approaches:

- **Direct Prompt** Leverage the LLM to directly generate judgment without fine-tuning.
- **SFT (w/o CoT)** Train a generative model by supervised fine-tuning to perform judgment without thinking traces.
- **SFT (w/ CoT)** Train a generative model on correct judgment thinking traces and results.

We also compare Think-J with the following classifier-based LLM-judge approaches:

- **BT Classifier** Feed the instruction and responses into the model and added a classification head, training it according to Bradley-Terry model (Sun, Shen, and Ton 2025).
- **CLoud** (Ke et al. 2024) First train the model to generate critiques, and then leverage the critiques as additional input to improve the BT Classifier.
- **SynRM** (Ye et al. 2024) Leverage critiques generated by a proprietary model as additional input to improve the BT Classifier. We use Deepseek-R1 to generate the critiques.

To further showcase Think-J's evaluation capabilities, we also compared its 32B version against leading proprietary judge models, including closed-source options like Claude 3.5 Sonnet, GPT-4o, and Gemini 1.5 Pro, as well as open-source fine-tuned judges such as JudgeLM, Prometheus,

[6]We do not include a length penalty in rewards to encourage longer thinking, as we have observed that longer thinking does not necessarily lead to better accuracy in our case.

[7]For example, a strength of 1 means the chosen response is only slightly better than the rejected, while a strength of 3 means the chosen is much better than the rejected. For more details about the criteria for the strength annotation, please refer to Appendix B.4.

[8]Please refer to Appendix A.1 for more details .

[9]huggingface.co/datasets/nvidia/HelpSteer2

[10]huggingface.co/datasets/Anthropic/hh-rlhf

Model	Sample Num	RewardBench					RMBench	Auto-J
		Chat	Hard	Safety	Reason	Overall	Overall	Agreement
Claude-3-5-Sonnet-20240620	—	96.4	74.0	81.6	84.7	84.2	68.9	70.7
Qwen-2.5-32B-Instruct	—	96.2	74.0	88.7	86.9	86.5	68.3	59.6
GPT-4o-2024-08-06	—	96.1	76.1	88.1	86.6	86.7	68.8	69.8
Gemini-1.5-Pro-0514	—	92.3	80.6	87.9	92.0	88.2	74.4	68.1
JudgeLM-33B (Zhu, Wang, and Wang 2023)	100K	90.1	51.0	85.7	39.7	66.6	49.6	45.3
Prometheus-7b-v2.0 (Zhu et al. 2023)	40K	83.9	49.2	72.8	72.0	69.5	52.4	63.1
Prometheus-8x7b-v2.0 (Zhu et al. 2023)	40K	93.0	47.1	80.5	77.4	74.5	57.4	68.5
Llama-3-OffsetBias-8B (Park et al. 2024)	276K	92.5	80.3	86.8	76.4	84.0	66.0	68.7
CompassJudger-32B (Cao et al. 2024)	2041K	97.4	65.6	85.1	87.1	83.8	69.4	**80.7**
STE-Llama3.1-70B (Wang et al. 2024c)	20K	96.9	85.1	89.6	88.4	90.0	65.3	72.0
SynRM-Command-R-35B (Ye et al. 2024)	5K	97.5	76.8	88.5	86.3	87.3	—	—
CLoud-Llama3-70B (Ke et al. 2024)	350K	98.0	75.6	87.6	89.0	87.6	—	—
Think-J-Qwen-2.5-32B (Helpsteer2-Pref)	9.8K	96.7	83.2	90.1	92.0	**90.5**	**79.8**	**75.8**

Table 2: Experiment results of top LLM-Judges on RewardBench, RMBench, and Auto-J-test. In this table, we report the best performance achieved by various LLM-Judges, trained on different base models and datasets.

Model	Method	HH-RLHF RewardBench					Helpsteer2-Pref RewardBench				
		Chat	Hard	Safety	Reason	Overall	Chat	Hard	Safety	Reason	Overall
Llama3-8B -Instruct	Direct Prompt (w/o CoT)	90.4	44.7	76.5	63.4	68.8	90.4	44.7	76.5	63.4	68.8
	Direct Prompt (w/ CoT)	84.6	40.9	50.8	58.8	58.8	84.6	40.9	50.8	58.8	58.8
	SFT on LIMJ707	91.8	67.1	83.0	64.2	76.5	91.8	67.1	83.0	64.2	76.5
	SFT (w/o CoT)	88.1	50.6	81.0	69.1	72.2	88.0	41.1	41.5	47.8	54.6
	SFT (w/ CoT)	78.8	67.0	81.0	62.1	72.2	84.5	71.3	80.8	62.6	74.8
	BT Classifier	81.8	72.6	79.3	85.2	79.7	88.6	73.9	81.2	91.8	83.9
	CLoud	87.4	69.3	87.3	77.6	80.4	93.3	74.8	80.8	73.7	80.6
	SynRM	87.2	74.3	81.1	80.2	80.7	91.9	68.4	80.4	87.9	82.2
	Think-J	92.2	71.7	84.8	75.6	**81.1**	93.9	74.3	90.3	77.8	**84.1**
Qwen2.5-7B -Instruct	Direct Prompt (w/o CoT)	94.4	56.9	81.0	77.3	77.4	94.4	56.9	81.0	77.3	77.4
	Direct Prompt (w/ CoT)	93.0	58.1	81.2	77.8	77.5	93.0	58.1	81.2	77.8	77.5
	SFT on LIMJ707	89.5	75.2	80.7	74.5	80.0	89.5	75.2	80.7	74.5	80.0
	SFT (w/o CoT)	89.8	74.3	82.5	61.3	77.0	95.0	75.9	87.2	75.8	83.5
	SFT (w/ CoT)	94.1	74.0	86.0	64.7	79.7	88.7	69.7	84.5	76.1	79.8
	BT Classifier	82.4	69.7	87.0	76.9	79.0	95.0	67.8	85.0	61.4	77.3
	CLoud	85.2	79.2	86.6	57.9	77.2	94.7	73.3	82.3	59.0	77.3
	SynRM	90.5	65.1	77.8	70.6	76.0	96.7	61.4	82.8	61.4	75.6
	Think-J	94.4	70.2	83.8	79.8	**82.0**	96.1	78.6	85.9	80.4	**85.3**

Table 3: Experiment results of different LLM judge training methods on RewardBench. In this table, we report the performance of various methods trained on the same base models (Llama3-8B-Instruct and Qwen2.5-7B-Instruct) and datasets (HH-RLHF and Helpsteer2-Pref) to enable a more direct comparison.

and CompassJudger. These models are widely employed as LLM-as-a-Judge across diverse tasks.

We mainly perform evaluation on RewardBench (Lambert et al. 2024). We also evaluate our model on RMBench (Liu et al. 2025b) and Auto-J-test (Li et al. 2023a).

We report the best results achieved by either offline or online learning for the main experiments by default.

4.2 Main Experiment

As demonstrated in Table 2, Think-J-32B achieved the best performance across all benchmarks, surpassing both close-sourced and fine-tuned judges. Notably, our method required only 9832 training samples, but still achieve marginal improvement on 32B-sized models. This underscores the effectiveness of Think-J, which leverages thinking optimization with the correctness of judgment as feedback to enhance preference modeling capability.

Furthermore, as Table 3 illustrates, starting from the same base model and training data, our proposed method consistently outperforms other approaches for fine-tuning LLM judges. This demonstrates the effectiveness of our judgment thinking optimization. In contrast, both naive and fine-tuned generative methods yield inferior results. Notably, the rejection sampling method, despite being trained on the same data constructed by the critic, also underperforms. This underscores the critical role of learning from negative samples when modeling human preference (Liu et al. 2024b).

On the other hand, the classifier-based method achieves relatively higher results but can only produce numerical outputs that lack interpretability. Additionally, the reasoning-enhanced classifiers, including CLoud and SynRM, performs worse than expected. This suggests that combining generative CoT into classifier may introduce noise rather than useful information for classification.

Method	Llama-3-8B-Instruct RewardBench					Qwen-2.5-7B-Instruct RewardBench				
	Chat	Hard	Safety	Reason	Overall	Chat	Hard	Safety	Reason	Overall
baseline	90.4	44.7	76.5	63.4	68.8	94.4	56.9	81.0	77.3	77.4
offline (Critic-guided)	95.0	73.3	87.4	77.2	83.2	94.8	74.2	83.5	80.5	83.3
offline (w/o SFT)	95.1	63.8	88.3	77.6	81.2	95.7	73.8	86.0	74.6	82.5
online (PPO)	70.4	75.3	77.2	70.0	73.2	88.7	69.7	84.5	76.1	79.8
online (Reinforce++)	93.7	74.3	89.7	77.9	84.0	94.7	70.6	90.4	82.3	84.5
online (GRPO)	93.9	74.3	90.3	77.8	**84.1**	96.1	78.6	85.9	80.4	**85.3**

Table 4: Experiment results of different RL strategies.

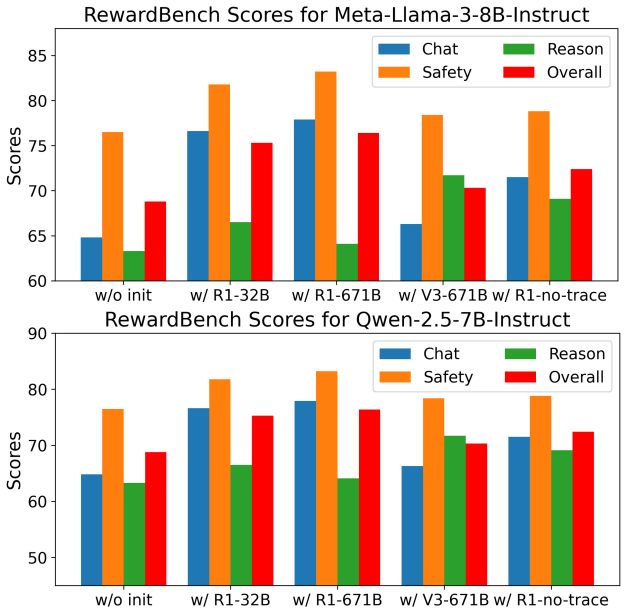

Figure 3: Experiment results of different data source for judgment thinking initialization.

Method	RewardBench			
	Chat	Safety	Reason	Overall
Llama-3-8B-Inst	64.8	76.5	63.4	68.8
Init with 1000 samples on chatbot-arena				
random-sampled	66.0	74.3	64.4	68.3
Init with 1000 samples on skywork-preference-v3				
random-sampled	75.1	82.0	66.4	75.4
longest	77.2	79.3	61.8	74.2
Llama3-failed	76.4	85.0	66.5	**76.1**

Table 5: Experiment results of different data selection strategies for judgment thinking initialization.

Init	RL	RewardBench			
		Chat	Safety	Reason	Overall
Llama-3-8B-Inst		64.8	76.5	63.4	68.8
no init	offline	80.3	73.2	79.4	79.1
	online	80.8	86.8	71.8	80.8
w/ init	offline	82.8	87.4	77.2	83.2
	online	82.9	90.3	77.8	**84.1**

Table 6: Experiment results of different initialization strategies for RL optimization on HelpSteer2-Pref.

5 Analysis

5.1 Less is More for Thinking Initialization

We compared the performance of different data source for judgment thinking initialization in Figure 3. Our findings indicate that a small amount of thinking trace annotated with Deepseek-R1 can significantly enhance the model's judgment capabilities. In contrast, using thinking trace annotated with Deepseek-V3, or removing the trace from the data would results in a substantial decline in performance. This highlights the importance of high-quality thinking trace for judgment thinking initialization.

We also compare the impact of different data selection strategies in Table 5. The results show that data quality is crucial for effective model initialization. For instance, a model initialized with chatbot-arena(Chiang et al. 2024), which contains relatively noisy data, achieves minimal improvement. Conversely, selecting the longest traces proves detrimental, as the longest traces are often code or math-related, which can negatively impact the data diversity.

We further investigated the impact of different initialization strategies on the subsequent optimization process, as shown in Table 6. The results indicate that even without thinking initialization, RL-based methods can still achieve substantial improvements, validating their effectiveness. Moreover, initializing the model with a few R1-annotated traces leads to a more structured and effective reasoning pattern, resulting in further enhancements in performance.

5.2 Best Practice for Thinking Optimization

With new RL algorithms continue to emerge in LLM training (Zhang et al. 2025), in this section, we compare different RL strategies for judgment thinking optimization.

As shown in Table 4, for offline learning, removing SFT term will lead to performance degradation, as SFT is conducted on token-level and can provide regularization for better preference optimization. On the other hand, for online learning, we find that PPO (Schulman et al. 2017) significantly underperforms compared to GRPO and Reinforce++ (Hu, Liu, and Shen 2025). This discrepancy suggests that incorporating a value model for thinking optimization would decrease training stability. We argue that natu-

Setting	Qwen2.5-32B-Instruct RewardBench					Qwen2.5-7B-Instruct RewardBench				
	Chat	Hard	Safety	Reason	Overall	Chat	Hard	Safety	Reason	Overall
β=1.0	96.7	83.2	90.1	92.0	**90.5**	96.1	78.6	85.9	80.4	**85.3**
β=0.5	96.9	85.8	91.2	87.4	90.3	95.0	77.3	87.6	77.2	84.3
β=0.0	92.9	71.3	86.3	67.3	79.4	95.2	71.5	88.0	71.7	81.6
γ=0.5	91.2	75.2	71.2	61.5	74.8	93.6	63.4	82.0	74.1	78.3
γ=0.2	96.7	83.2	90.1	92.0	**90.5**	94.7	76.3	88.7	78.2	84.5
γ=0.0	96.0	83.7	90.5	87.7	89.5	96.1	78.6	85.9	80.4	**85.3**

Table 7: Experiment results of different reward function settings on Helpsteer2-Pref.

Method	RewardBench			
	Chat	Safety	Reason	Overall
Llama-3-8B-Inst	64.8	76.5	63.4	68.8
Offline learning on Helpsteer2				
iteration 1	81.8	88.3	77.8	83.2
iteration 2	82.5	87.2	74.8	82.5
iteration 3	82.8	87.4	77.2	**83.2**
Offline learning on HH-RLHF				
iteration 1	81.2	83.2	67.0	77.3
iteration 2	81.1	84.6	70.5	78.4
iteration 3	78.3	83.1	72.6	**78.9**

Table 8: Experiment results of iterative offline learning.

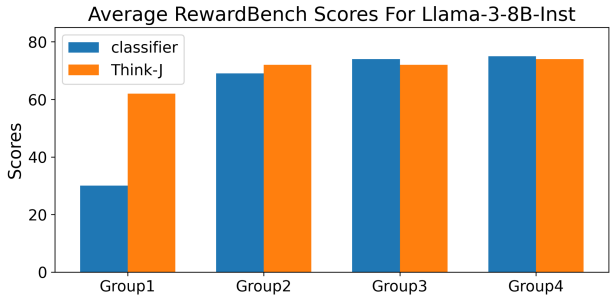

Figure 4: Comparison of different methods on data groups with different quality. Group 1 is with lower quality and Group 4 is with higher quality.

ral language generation (NLG) tasks is distinct from the sequential decision-making tasks in traditional RL. Therefore, the introduction of an additional value model is not only unnecessary but may also hinder training efficiency.

Finally, as shown in Table 8, offline learning in an iterative manner can achieve further improvement. However, this will result in a more complex and cumbersome training pipeline. To draw a conclusion, the best practice for thinking optimization is GRPO with carefully designed rewards.

5.3 Reward Design for Online Learning

In this section, we aim to analyze the contribution of different components of the reward function as defined in 6. We fixed α as 1.0 and varied the weights β, and γ.

As shown in the results in Table 7, the formatting reward r_{format} is crucial for both models, as its removal leads to significant performance degradation. Conversely, the influence of the strength reward r_{strength} differs between the models. For Qwen-2.5-7B-Inst, removing r_{strength} results in a performance improvement, while it offers a slight enhancement for the more capable 32B model. We hypothesize that 32B model's stronger inherent abilities allow it to effectively learn the correlation between preference strength and judgment. In contrast, for the weaker 7B model, this additional learning objective may introduce confusion.

5.4 Thinking Makes the Judge More Robust

In real applications, it is common that there exists noise in the training set, or there is a distributional difference between the training and test sets. In such cases, Think-J demonstrates superior robustness compared with classifiers as a result of its judgment thinking ability.

To verify this, we adopted the approach from (Wang et al. 2024b) and divided HH-RLHF into four groups with different data quality[11]. We then trained judges on the data based on classifier-judge or Think-J. As shown in Figure 4, for the two groups with higher data quality, classifier-based judge achieve comparable or even better performance. However, for the groups with lower quality, the accuracy of classifier-based methods drops significantly, even falling below random guessing. In contrast, Think-J maintains relative stability, verifying its robustness to varied data quality[12].

6 Conclusion

In this paper, we propose Think-J to enhance generative LLM-Judges with judgment thinking optimization. Experiment results verify the effectiveness of Think-J compared with both classifier-based and generative LLM judges. With the increasing popularity of RL-based test-time scaling methods, it is crucial to develop a reliable and stable feedback system that aligns well with real-world human preferences. In future, we will continue to explore generative judges for more accurate preference modeling.

[11]Data quality is indicated by the difference in scores assigned to response pairs by an external reward model. For more details please refer to the work of (Wang et al. 2024b).

[12]We also present thinking trace error analysis in Appendix A.6.

Acknowledgements

This work was supported in part by the Jiangsu Science and Technology Major Project (BG2024031) and Nanjing University AI & AI for Science Funding (2024300540).

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Cao, M.; Lam, A.; Duan, H.; Liu, H.; Zhang, S.; and Chen, K. 2024. CompassJudger-1: All-in-one Judge Model Helps Model Evaluation and Evolution. arXiv:2410.16256.

Chang, Y.; Wang, X.; Wang, J.; Wu, Y.; Yang, L.; Zhu, K.; Chen, H.; Yi, X.; Wang, C.; Wang, Y.; Ye, W.; Zhang, Y.; Chang, Y.; Yu, P. S.; Yang, Q.; and Xie, X. 2023. A Survey on Evaluation of Large Language Models. arXiv:2307.03109.

Chen, N.; Hu, Z.; Zou, Q.; Wu, J.; Wang, Q.; Hooi, B.; and He, B. 2025a. JudgeLRM: Large Reasoning Models as a Judge. arXiv:2504.00050.

Chen, X.; Li, G.; Wang, Z.; Jin, B.; Qian, C.; Wang, Y.; Wang, H.; Zhang, Y.; Zhang, D.; Zhang, T.; Tong, H.; and Ji, H. 2025b. RM-R1: Reward Modeling as Reasoning. arXiv:2505.02387.

Chiang, W.-L.; Zheng, L.; Sheng, Y.; Angelopoulos, A. N.; Li, T.; Li, D.; Zhang, H.; Zhu, B.; Jordan, M.; Gonzalez, J. E.; and Stoica, I. 2024. Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference. arXiv:2403.04132.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. *arXiv e-prints*, arXiv–2407.

Gu, J.; Jiang, X.; Shi, Z.; Tan, H.; Zhai, X.; Xu, C.; Li, W.; Shen, Y.; Ma, S.; Liu, H.; et al. 2024. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

He, Y.; Li, S.; Liu, J.; Wang, W.; Bu, X.; Zhang, G.; Peng, Z.; Zhang, Z.; Zheng, Z.; Su, W.; et al. 2025. Can large language models detect errors in long chain-of-thought reasoning? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 18468–18489.

Hu, J.; Liu, J. K.; and Shen, W. 2025. REINFORCE++: An Efficient RLHF Algorithm with Robustness to Both Prompt and Reward Models. arXiv:2501.03262.

Huang, H.; Bu, X.; Zhou, H.; Qu, Y.; Liu, J.; Yang, M.; Xu, B.; and Zhao, T. 2025. An Empirical Study of LLM-as-a-Judge for LLM Evaluation: Fine-tuned Judge Model is not a General Substitute for GPT-4. In Che, W.; Nabende, J.; Shutova, E.; and Pilehvar, M. T., eds., *Findings of the Association for Computational Linguistics: ACL 2025*, 5880–5895. Vienna, Austria: Association for Computational Linguistics. ISBN 979-8-89176-256-5.

Huang, H.; Qu, Y.; Bu, X.; Zhou, H.; Liu, J.; Yang, M.; Xu, B.; and Zhao, T. 2024. An Empirical Study of LLM-as-a-Judge for LLM Evaluation: Fine-tuned Judge Model is not a General Substitute for GPT-4. arXiv:2403.02839.

Jaech, A.; Kalai, A.; Lerer, A.; Richardson, A.; El-Kishky, A.; Low, A.; Helyar, A.; Madry, A.; Beutel, A.; Carney, A.; et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Ke, P.; Wen, B.; Feng, A.; Liu, X.; Lei, X.; Cheng, J.; Wang, S.; Zeng, A.; Dong, Y.; Wang, H.; Tang, J.; and Huang, M. 2024. CritiqueLLM: Towards an Informative Critique Generation Model for Evaluation of Large Language Model Generation. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 13034–13054. Bangkok, Thailand: Association for Computational Linguistics.

Kim, S.; Shin, J.; Cho, Y.; Jang, J.; Longpre, S.; Lee, H.; Yun, S.; Shin, S.; Kim, S.; Thorne, J.; et al. 2023. Prometheus: Inducing Fine-grained Evaluation Capability in Language Models. *arXiv preprint arXiv:2310.08491*.

Lambert, N.; Pyatkin, V.; Morrison, J.; Miranda, L.; Lin, B. Y.; Chandu, K.; Dziri, N.; Kumar, S.; Zick, T.; Choi, Y.; Smith, N. A.; and Hajishirzi, H. 2024. RewardBench: Evaluating Reward Models for Language Modeling. arXiv:2403.13787.

Li, J.; Sun, S.; Yuan, W.; Fan, R.-Z.; Zhao, H.; and Liu, P. 2023a. Generative Judge for Evaluating Alignment. *arXiv preprint arXiv:2310.05470*.

Li, X.; Zhang, T.; Dubois, Y.; Taori, R.; Gulrajani, I.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023b. AlpacaEval: An Automatic Evaluator of Instruction-following Models. https://github.com/tatsu-lab/alpaca_eval.

Liu, C. Y.; Zeng, L.; Liu, J.; Yan, R.; He, J.; Wang, C.; Yan, S.; Liu, Y.; and Zhou, Y. 2024a. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*.

Liu, T.; Zhao, Y.; Joshi, R.; Khalman, M.; Saleh, M.; Liu, P. J.; and Liu, J. 2024b. Statistical Rejection Sampling Improves Preference Optimization. In *The Twelfth International Conference on Learning Representations*.

Liu, W.; He, Y.; Li, Y.; Huang, H.; Hu, C.; Liu, J.; Li, S.; Su, W.; and Zheng, B. 2025a. Air: Complex instruction generation via automatic iterative refinement. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 31952–31974.

Liu, Y.; Yao, Z.; Min, R.; Cao, Y.; Hou, L.; and Li, J. 2025b. RM-Bench: Benchmarking Reward Models of Language Models with Subtlety and Style. In *The Thirteenth International Conference on Learning Representations*.

Liu, Z.; Liu, J.; He, Y.; Wang, W.; Liu, J.; Pan, L.; Hu, X.; Xiong, S.; Huang, J.; Hu, J.; Huang, S.; Obando-Ceron, J.;

Yang, S.; Wang, J.; Su, W.; and Zheng, B. 2025c. Part I: Tricks or Traps? A Deep Dive into RL for LLM Reasoning. arXiv:2508.08221.

Liu, Z.; Wang, P.; Xu, R.; Ma, S.; Ruan, C.; Li, P.; Liu, Y.; and Wu, Y. 2025d. Inference-Time Scaling for Generalist Reward Modeling. arXiv:2504.02495.

Muennighoff, N.; Yang, Z.; Shi, W.; Li, X. L.; Fei-Fei, L.; Hajishirzi, H.; Zettlemoyer, L.; Liang, P.; Candès, E.; and Hashimoto, T. 2025. s1: Simple test-time scaling. arXiv:2501.19393.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In Isabelle, P.; Charniak, E.; and Lin, D., eds., *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 311–318. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics.

Park, J.; Jwa, S.; Meiying, R.; Kim, D.; and Choi, S. 2024. OffsetBias: Leveraging Debiased Data for Tuning Evaluators. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Findings of the Association for Computational Linguistics: EMNLP 2024*, 1043–1067. Miami, Florida, USA: Association for Computational Linguistics.

Qwen; :; Yang, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; Lin, H.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Lin, J.; Dang, K.; Lu, K.; Bao, K.; Yang, K.; Yu, L.; Li, M.; Xue, M.; Zhang, P.; Zhu, Q.; Men, R.; Lin, R.; Li, T.; Tang, T.; Xia, T.; Ren, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; and Qiu, Z. 2025. Qwen2.5 Technical Report. arXiv:2412.15115.

Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C. D.; Ermon, S.; and Finn, C. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y. K.; Wu, Y.; and Guo, D. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv:2402.03300.

Sun, H.; Shen, Y.; and Ton, J.-F. 2025. Rethinking Bradley-Terry Models in Preference-Based Reward Modeling: Foundations, Theory, and Alternatives. arXiv:2411.04991.

Wang, B.; Zheng, R.; Chen, L.; Liu, Y.; Dou, S.; Huang, C.; Shen, W.; Jin, S.; Zhou, E.; Shi, C.; Gao, S.; Xu, N.; Zhou, Y.; Fan, X.; Xi, Z.; Zhao, J.; Wang, X.; Ji, T.; Yan, H.; Shen, L.; Chen, Z.; Gui, T.; Zhang, Q.; Qiu, X.; Huang, X.; Wu, Z.; and Jiang, Y.-G. 2024a. Secrets of RLHF in Large Language Models Part II: Reward Modeling. arXiv:2401.06080.

Wang, B.; Zheng, R.; Chen, L.; Xi, Z.; Shen, W.; Zhou, Y.; Yan, D.; Gui, T.; Zhang, Q.; and Huang, X. 2024b. Reward Modeling Requires Automatic Adjustment Based on Data Quality. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Findings of the Association for Computational Linguistics: EMNLP 2024*, 4041–4064. Miami, Florida, USA: Association for Computational Linguistics.

Wang, T.; Kulikov, I.; Golovneva, O.; Yu, P.; Yuan, W.; Dwivedi-Yu, J.; Pang, R. Y.; Fazel-Zarandi, M.; Weston, J.; and Li, X. 2024c. Self-Taught Evaluators. arXiv:2408.02666.

Wang, Y.; Li, Z.; Zang, Y.; Wang, C.; Lu, Q.; Jin, C.; and Wang, J. 2025a. Unified Multimodal Chain-of-Thought Reward Model through Reinforcement Fine-Tuning. arXiv:2505.03318.

Wang, Z.; Bukharin, A.; Delalleau, O.; Egert, D.; Shen, G.; Zeng, J.; Kuchaiev, O.; and Dong, Y. 2024d. HelpSteer2-Preference: Complementing Ratings with Preferences. arXiv:2410.01257.

Wang, Z.; Zeng, J.; Delalleau, O.; Egert, D.; Evans, E.; Shin, H.-C.; Soares, F.; Dong, Y.; and Kuchaiev, O. 2025b. Dedicated Feedback and Edit Models Empower Inference-Time Scaling for Open-Ended General-Domain Tasks. arXiv:2503.04378.

Whitehouse, C.; Wang, T.; Yu, P.; Li, X.; Weston, J.; Kulikov, I.; and Saha, S. 2025. J1: Incentivizing Thinking in LLM-as-a-Judge via Reinforcement Learning. arXiv:2505.10320.

Ye, Y.; Huang, Z.; Xiao, Y.; Chern, E.; Xia, S.; and Liu, P. 2025. LIMO: Less is More for Reasoning. arXiv:2502.03387.

Ye, Z.; Greenlee-Scott, F.; Bartolo, M.; Blunsom, P.; Campos, J. A.; and Gallé, M. 2024. Improving Reward Models with Synthetic Critiques. arXiv:2405.20850.

Zhang, Q.; Lyu, F.; Sun, Z.; Wang, L.; Zhang, W.; Guo, Z.; Wang, Y.; Muennighoff, N.; King, I.; Liu, X.; and Ma, C. 2025. What, How, Where, and How Well? A Survey on Test-Time Scaling in Large Language Models. arXiv:2503.24235.

Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E.; et al. 2023. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *arXiv preprint arXiv:2306.05685*.

Zhu, K.; Zhao, Q.; Chen, H.; Wang, J.; and Xie, X. 2023. PromptBench: A Unified Library for Evaluation of Large Language Models. *arXiv preprint arXiv:2312.07910*.

Zhu, L.; Wang, X.; and Wang, X. 2023. Judgelm: Fine-tuned large language models are scalable judges. *arXiv preprint arXiv:2310.17631*.