Meta Ranking: Less Capable Language Models are Capable for Single Response Judgement

Anonymous ACL submission

Abstract

Although Large Language Models (LLMs) have demonstrated strong performance on a wide range of tasks, they still face reliability challenges such as hallucination. Previous studies reveal that highly capable LLMs like GPT-4 are effective in judging the reliability of individual responses while less capable ones are often tuned to evaluate the relative reliability of responses to the same query. To enable less capable LLMs to effectively judge the reliability of individual responses, we propose a novel method named *Meta Ranking* (MR). Unlike previous methods, which assess the response directly, we achieve the judgement by comparing the target query-response pair with reference query-response pairs. We found its remarkable effectiveness in error detection for LLM responses on reasoning tasks, where less capable LLMs could outperform strong baselines, even without fine-tuning. We further demonstrate that MR can be used to enhance the performance of LLMs in two practical applications: query routing and iterative training data filtering. The former achieves GPT-4-turbo comparable performance with less than half the token consumption, while the latter makes the instruction-tuned LLaMA-7B and Phi-2, a 2.7B model, significantly surpasses Alpaca-13B over fewer training samples, underscoring the high potential of our proposed method.

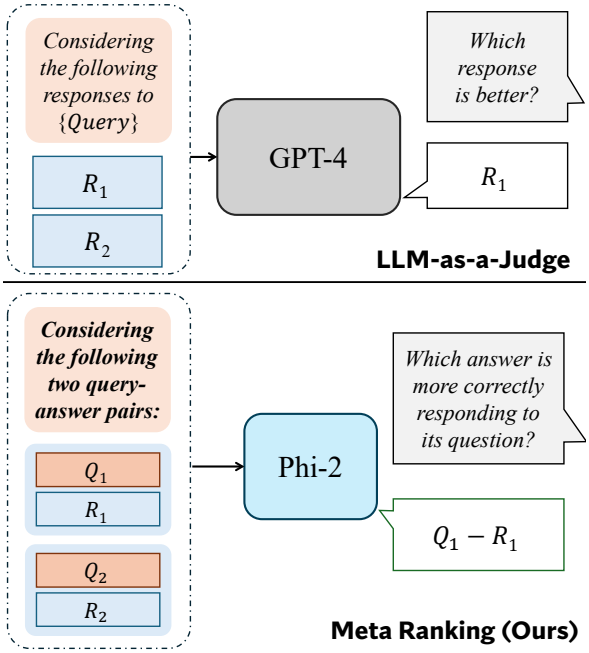

Figure 1: The illustration of the difference between LLM-as-a-Judge and our proposed *Meta Ranking* (MR) methods. The former contrasts responses of the same query. Instead, MR takes two query-response pairs for cross-query comparison on reliability, thereby able to judge single responses to their own queries. Q_i and R_i denote the query and response, respectively.

1 Introduction

Large language models (LLMs) have demonstrated strong performance in various tasks (OpenAI, 2023; Touvron et al., 2023b; Du et al., 2022). However, they still face reliability challenges. For example, these models often produce responses that seem plausible but are factually incorrect, a phenomenon known as "hallucination" (Huang et al., 2023). Fine-tuning and alignment techniques have been extensively studied to mitigate this issue (Ouyang et al., 2022; Wang et al., 2023; Rafailov et al., 2023). Recent studies sadly demonstrate that hallucination is inevitable (Xu et al., 2024). Consequently, instead of resolving the problems directly, we focus on developing techniques to discriminate the reliability of responses provided by LLMs.

Increasing research has demonstrated that highly capable or specifically fine-tuned LLMs can act as evaluators of response reliability (Zheng et al., 2023a). Highly capable models, such as GPT-4 (OpenAI, 2023), have proven effective in assessing the quality of individual responses. However, these models are often significantly large, leading to substantial computational and financial costs. In contrast, less capable models, when fine-tuned, are

more cost-effective and have shown to be effective in comparing the relative quality of multiple responses to the same query (Wang et al., 2024b; Ke et al., 2023; Zhu et al., 2023a). Nevertheless, these fine-tuned models are less effective at assessing the reliability of a single response, which restricts their application. Our experiments, as shown in Figure 3, support this finding. This raises the question: *How can we enable less capable LLMs to effectively judge the reliability of a single response?*

To address this question, we propose a novel method named *Meta Ranking* (MR). The basic assumption is that the reliability of a response to a query can be determined by comparing this query-response pair with other pairs whose reliabilities are already known. Unlike previous methods that let an LLM to directly judge one or multiple responses to the same query (Figure 1 (a)), our method compares the targeted query-response pair with reference pairs (Figure 1 (b)). Specifically, we use a fixed set of labeled query-response pairs as reference pairs. For a given target query-response pair, an LLM evaluates whether this pair is more reliable than each of the reference pairs. A voting-style mechanism is then employed to aggregate the comparison results to arrive at the final judgment. In this context, "reliable" refers to desired properties such as correctness and quality as required. Experimental results indicate that the MR method enables *small* LLMs to effectively judge the correctness of individual responses. Furthermore, we demonstrate that the MR method can enhance the performance of LLMs in two practical applications: (1) improving the efficiency of LLMs by routing queries to the appropriate LLMs based on the assessment of correctness, achieving performance comparable to GPT-4-turbo with less than half the token consumption; and (2) iteratively refining training datasets to enhance the performance of LLMs, leading to significant improvements over state-of-the-art data selection methods on the Alpaca-52k dataset (Taori et al., 2023).

In summary, our contributions are threefold:

1. We propose a novel method named *Meta Ranking* for accessing the reliability of LLM responses based on a novel mechanism: comparing the target query-response pair with the reference query-response pairs.

2. Experimental results reveal that MR enables *less capable* LLMs to judge the reliability of *individual* responses, which was achievable only with highly capable LLMs.

3. Additionally, we explore two practical applications of MR, significantly improving the efficiency and effectiveness of LLMs, respectively. These results underscore the considerable potential of our proposed method.

2 Meta Ranking

This section demonstrates how cross-query comparisons could reveal the reliability of the target query-response pair with limited labeled examples from the same source LLM. Below, we outline the specific steps and considerations accordingly.

Formally, suppose we have N reference query-response pairs

$$\mathcal{X} = \{(Q_i, R_i)\}\,, \tag{1}$$

where $i = 1, \cdots, N$, Q_i and R_i are the i-th reference query and response, respectively. Additionally, for each pair of (Q_i, R_i), we have a label C_i that represents its reliability. For binary classification scenarios (e.g., response correctness assessment), $C_i \in \{+1, -1\}$, where $+1$ denotes True and -1 denotes False. And for continuous metrics, $C_i \in \mathbb{R}$. We aim to evaluate the reliability of a target response Q_{target} to a target query Q_{target}.

The basic operation of *Meta Ranking* is to compare the target query-response pair with each of the reference query-response pairs. For brevity, we denote the target query-response pair as $S_{\text{target}} = (Q_{\text{target}}, R_{\text{target}})$, and the i-th query-response pair as $S_i = (Q_i, R_i)$. Then the comparison operation and result are denoted as follows:

$$r_i = \mathrm{MR}\left(S_{\text{target}}, S_i\right), \tag{2}$$

where $r_i \in \{\pm 1, 0\}$, $+1$, 0 and -1 denote the target pair is better than, equal to, or worse than the i-th reference pair, respectively. In practice, $\mathrm{MR}(\cdot, \cdot)$ is implemented by directly prompting LLMs, or using the relative magnitude of quality estimation scores of the query-response pairs.

The final judgement is obtained by a voting-style mechanism. The intuition is as follows: taking correctness assessment as an example, the target response is likely to be correct when it is ranked higher than a correct reference response, and likely to be incorrect otherwise, as shown in Figure 2. We aggregate the comparison results to arrive at the final judgement. Specifically, we will upvote the target response if the target query-response pair

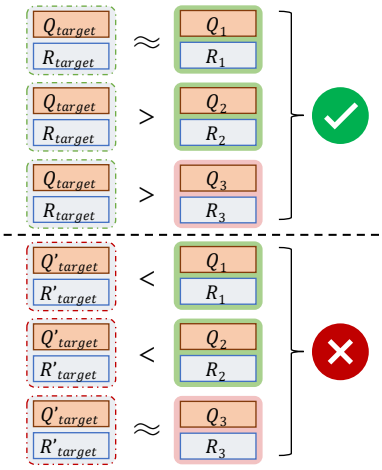

Figure 2: The few-shot *Meta Ranking* process. The correctness of the target response to the query is considered according to comparison results with labeled query-response pairs.

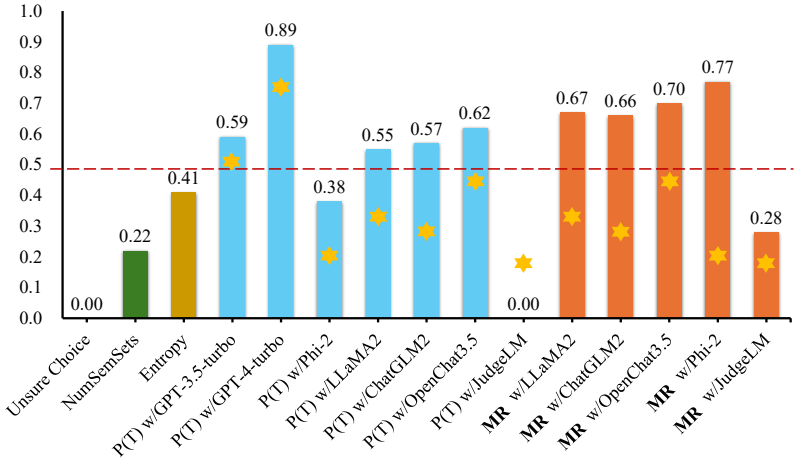

Figure 3: The micro precision scores of *Meta Ranking* and baselines in error detection experiments for responses from `LLaMA-2-chat-7B` on the MMLU dataset. The dashed red line represents the performance of random selection. We used examples in the development set for all few-shot methods. Stars denote the performance on MMLU of underlying LLMs of each method.

is ranked higher than the reference pair, i.e., $r_i = +1$, and downvote when ranked lower. Formally, we denote the total vote value as s, and the delta value Δs_i caused by the comparison with the i-th reference pair is calculated as follows:

$$\Delta s_i = C_i \cdot \delta_{\mathrm{sgn}(C_i) \cdot r_i}, \tag{3}$$

where $\mathrm{sgn}(\cdot)$ is the sign function. We can easily verify that $\mathrm{sgn}(C_i) \cdot r_i \in \{+1, 0, -1\}$, and δ_{+1}, δ_0 and δ_{-1} are hyperparameters. Note that we require that $\delta_{+1} > 0$ and $\delta_{-1} < 0$. Accordingly, we hav

$$s = \sum_{i=1}^{N} \Delta s_i, \tag{4}$$

and we say the target response is reliable if $s \geq 0$, and unreliable otherwise. The entire process is shown in Appendix B. Theoretically, when comparison results are accurate, we show that we have $\mathrm{sgn}(\Delta s_i) \equiv \mathrm{sgn}(C_{\text{target}} - C_i)$ under reasonable constraints in Appendix C.1. Thus, $\mathrm{sgn}(s)$ could be an approximation for $\mathrm{sgn}(C_{\text{target}} - C_{\text{avg}})$, where $C_{\text{avg}} = \frac{1}{N}\sum_i C_i$. Hence, a negative s means the subpar reliability of the target response, and vice versa. In practice, N is usually small due to efficiency and the limited labeled data.

3 Error Detection on Reasoning Problems

In the following section, we empirically demonstrate that *Meta Ranking* can accurately evaluate the reliability of a single response concerning correctness. We leverage error detection on responses generated by LLMs for reasoning tasks for validation. Our findings indicate that the MR approach maintains consistent efficacy across varying base models, response accuracies, and question languages.

3.1 Settings

The error detection task requires identifying whether a response from an LLM is incorrect given the query. The settings are as follows:

Datasets: We randomly select two subjects of each category in the MMLU dataset (Hendrycks et al., 2021b,a) and one subject of each category in the CMMLU dataset (Li et al., 2023a) as the Chinese benchmark. The datasets contain multi-choice questions from a variety of areas, with five examples for each subject in the development set.

Response Generation: For different accuracy levels, we choose `LLaMA-2-chat-7B` (Touvron et al., 2023b) and `OpenChat-3.5` (Wang et al., 2024a) to generate responses for English questions, and `ChatGLM-2-6B` (Zeng et al., 2023) and `Yi-6B-Chat` (01.AI, 2023) for Chinese reasoning problems. We then use the same model to generate responses for reference on the development set.

MR Settings: We prompted LLaMA-2, ChatGLM-2, OpenChat-3.5, and Phi-2 to judge on different query-response pairs for *Meta Ranking*. We also tested an LLM-as-a-Judge-tuned model `JudgeLM-7B-v1` (Zhu et al., 2023a) in MR to see if the fine-tuning for the evaluation of responses to the same query helps. By setting each label C_i with the value $\{\text{True}, \text{False}\} = \{1, -1\}$, we apply MR on this task by identifying incorrect responses depending on MR results on the query-response

Method	MMLU (En)		CMMLU (Zh)	
	LLaMA-2	OpenChat-3.5	ChatGLM-2	Yi
Random Selection	*0.50*	*0.50*	*0.50*	*0.50*
P(T) w/ OpenChat-3.5	0.62	0.38	0.31	0.35
P(T) w/ GPT-3.5-turbo	0.59	0.65	0.22	0.21
MR w/ OpenChat-3.5	0.70	0.63	**0.73**	**0.72**
MR w/ Phi-2	**0.77**	**0.73**	0.69	0.52

Table 1: The micro precision scores on error detection experiments on the MMLU and CMMLU datasets with responses generated by different LLMs. The **bold font** denotes best results. LLMs in the second row of the header are sources of responses.

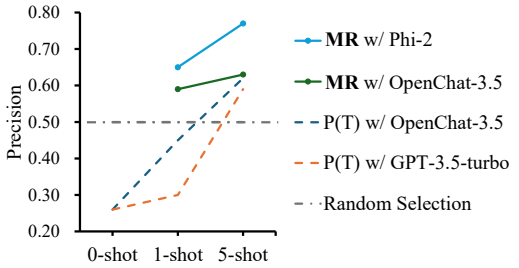

Figure 4: The change of precision scores with the number of reference pairs on the MMLU dataset with responses from LLaMA-2.

pairs according to Algorithm 1.

Baselines: We compare our method against several baselines to ensure a comprehensive evaluation, including appending an (1) **Unsure Choice**, a black-box uncertainty estimation method (2) **NumSemSets** (Lin et al., 2023) and a white-box method (3) **Entropy** (Han et al., 2024), as well as (4) P(True) (**P(T)**) (Kadavath et al., 2022) which directly asks an LLM about the correctness of a query-response pair. Implementation details are in Appendix B.1.

3.2 Discussion

Effectiveness of MR In Figure 3, we report micro precision scores of all baselines and MR in error detection on LLaMA-2-generated responses and the actual performance of LLMs on MMLU. Impressively, we found that MR with Phi-2 notably exceeds all baseline methods, except for P(T) with GPT-4-turbo, reaching a precision score of 0.77, more than double the performance of P(T) with Phi-2 and reaching 88% GPT-4-turbo performance. With LLaMA-2 and ChatGLM-2, MR exceeds P(T) with the same model for 22% and 16% in precision scores. However, JudgLM performs not as well as other pretrained or general instruction-tuned LMs in the MR results and completely fails to generalize for the P(T) method. We also depict F1 scores in Appendix B.1 to indicate MR is not biased to identify most responses as incorrect ones. We also notice that uncertainty-based methods NumSemSets and Entropy performs worse than random selection, potentially indicating errors in reasoning paths are difficult to distinguish by uncertainty scores from the LLM used in generation. It could also result from the lack of reference pairs. In the contrast, MR consistently outperforms the random baseline to a greater margin than P(T), except for P(T) with GPT-4-turbo. GPT-4-turbo could accurately detect the error, probably relying on its strong reasoning capabilities and generalizability (OpenAI, 2023).

Effectiveness of Less Capable Language Models In Figure 3, the performance of P(T) across different models displays a positive correlation with their actual performance on reasoning tasks, while MR has demonstrated strong robustness across models with different capabilities. We further investigate the performance of LLMs with different capabilities across languages and tasks. In Table 1, we select top performing P(T) and MR methods and report the precision score across all responses on the MMLU and CMMLU datasets. We omit GPT-4-turbo for the cost and its unparalleled capabilities to open-source models we have tested. From results, P(T) with OpenChat-3.5 performs worse than random selection when facing more accurate responses on MMLU. In contrast, MR shows significant generalizability with less capable models, e.g., 2.7B Phi-2, across different accuracy levels of responses, impressively surpassing P(T) w/ GPT-3.5-turbo and OpenChat-3.5 on responses in different accuracy levels. It positively indicates the effectiveness of less capable LLMs in accurately detecting reasoning errors in single responses.

Impact of the Number of Reference Pairs In Figure 4, we illustrate the change of precision scores with the number of reference pairs, where MR exhibits its robustness compared to P(T). Notably, other uncertainty-based methods are incompatible with the 1-shot setting since there are usually no correct examples for calibration. Upon the ablation study, we observe that reducing five examples to one leads to a slightly decreased performance of MR, indicating the robustness of the cross-query comparison. Also, the result highlights the effectiveness of MR with limited labeled data compared to P(T). Without reference examples, P(T) faces a great performance drop even with capable models (e.g., GPT-3.5-turbo and OpenChat-3.5), showing inferior performance to random selection. It is also worth noting that uncertainty-based methods

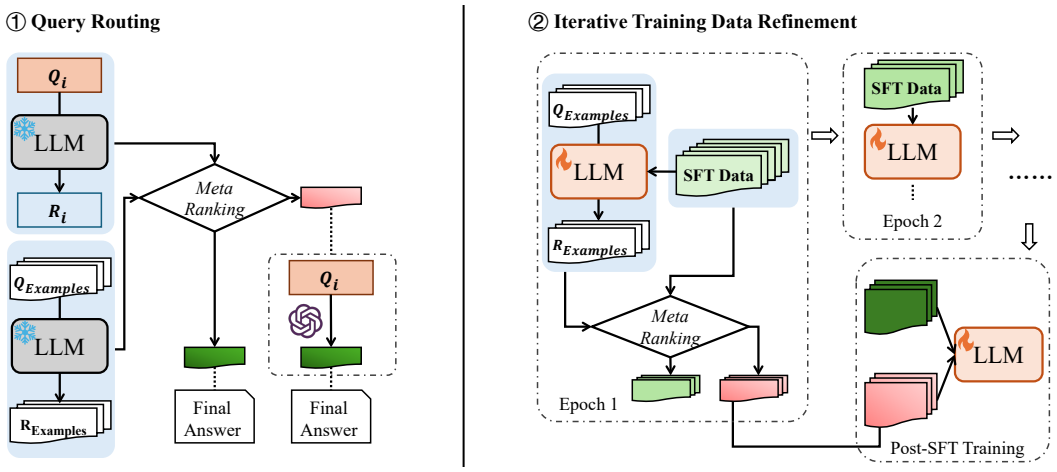

Figure 5: Two applications of *Meta Ranking* for inference- and training-time LLM enhancement, respectively. (1) Query Routing: identify and route unsolved queries from open-source LLMs towards API-based LLMs for better results. (2) Iterative Training Data Refinement: filter low-quality data after each epoch in SFT, and tune LLMs with low- and high-quality data at the second stage. MR results depend on the reference pairs generated from the LLM after each epoch. $Q_{\text{Examples}}, R_{\text{Examples}}$ denote the labeled reference pairs for the MR algorithm (Algorithm 1).

like NumSemSets and Entropy usually require hundreds of examples for calibration on distinguishing errors (Han et al., 2024; Mielke et al., 2022), explaining the relative low results in Figure 3 when there are only five examples for reference.

Performance on Non-English Tasks In Table 1, the overall results on Chinese reasoning problems are significantly lower than on English ones, demonstrating that non-English languages do influence error detection performance. However, MR exhibits strong robustness across languages, though P(T) performs worse than random selection (0.5 for precision scores) in all CMMLU results. Please see Appendix D.1 for experiments on Japanese as a representative low-resource language.

4 Applications

In this section, we present two promising applications to further validate the effectiveness of *Meta Ranking*, as shown in Figure 5. (1) With the assessment of correctness on single responses to the given queries, we can enhance the capability of open-source LLMs during inference, by identifying and routing unsolved queries to stronger API-based LLMs. It has the potential for achieving better efficiency and remaining performance of API-based models. (2) Moreover, we can refine supervised fine-tuning (SFT) process for LLM alignment, whose key factor is the quality of training data from recent studies (Liu et al., 2024). By filtering low-quality data after each epoch and introducing post-SFT training with mere instruction data, we achieve significant improvement against state-of-the-art SFT data selection methods.

4.1 Query Routing

4.1.1 Implementation

Since LLMs exhibit varying degrees of confidence and accuracy across different types of queries (Yin et al., 2023), we propose to use MR on a query routing mechanism. As shown in Figure 5 ②, This mechanism adopts MR to assess the reliability of generated responses of open-source LLMs. Queries with responses for which MR decides to be unreliable in terms of correctness are routed to API-based LLMs, which are typically more powerful but also more costly, for better answers. Through this mechanism, we aims to achieve similar performance to API-based models with better efficiency.

Given that the MR method requires labeled query-response examples, we first feed the queries in the development set to the open-source LLM and evaluate the generated responses with the groundtruth. Formally, let us consider two sets of queries: the test set $\mathcal{X}_{\text{test}}$, which consists of N_{test} items denoted as $\{Q_i\}_{i=1}^{N_{\text{test}}}$, and the development set $\mathcal{X}_{\text{dev}}$, comprising N_{dev} items denoted as $\{Q_i\}_{i=1}^{N_{\text{dev}}}$. Both sets are assumed to be drawn from the same underlying distribution. For every query Q_i, we use $R_i^{(\theta)}$ to represent the response generated by an open-source LLM (parameterized by θ), and $C_i^{(\theta)}$ to denote the evaluation result according to the groundtruth and a proper metric. By applying the model to each query in $\mathcal{X}_{\text{dev}}$, we can produce responses $\left\{R_i^{(\theta)}\right\}_{i=1}^{N_{\text{dev}}}$ and form a set of reference query-response pairs

Model	Routing Strategy	Reasoning		Translation		Average	#Token (API) (Relative Value)
		English	Chinese	Zh-En	En-Zh		
Phi-2	-	22.48	24.84	42.32	23.18	28.21	-
LLaMA-2 (①)	-	34.37	32.86	58.53	51.33	44.27	-
ChatGLM-2 (②)		30.48	49.71	58.43	63.30	50.48	
`Ensemble` (①&②)		35.44	43.47	31.35	46.54	39.20	
GPT-3.5-turbo (③)		52.91	54.09	63.67	69.14	59.95	1.00
①/② + ③	`Entropy`	38.75	47.03	58.32	65.45	52.39 (+1.91)	0.38
	`Random`	41.89	48.32	58.97	64.83	53.50 (+3.02)	0.45
	MR$_\Delta$	44.30	48.67	59.80	65.87	54.66 (+4.18)	**0.24**
	MR	**48.62**	**52.76**	**61.36**	**67.10**	**57.46** (**+6.98**)	0.42
OpenChat-3.5 (④)	-	45.42	40.19	61.35	60.77	51.93	-
Yi (⑤)		42.08	61.96	60.87	62.07	56.74	
`Ensemble` (④&⑤)		45.68	48.24	11.16	62.90	41.99	
GPT-4-turbo (⑥)		72.86	62.82	64.73	69.95	67.59	1.00
④/⑤ + ⑥	`Random`	46.64	**61.96**	61.29	63.74	58.41 (+1.67)	0.44
	MR$_\Delta$	57.68	61.93	61.60	67.61	62.21 (+5.47)	**0.23**
	MR	**64.68**	61.93	**62.60**	**68.11**	**64.33** (**+7.59**)	0.43

Table 2: The overall results in query routing experiments. All routing strategies are implemented with `Phi-2`. "Overall" denotes the macro average value across tasks. The **bold font** denotes the best result using query routing and the underlined numbers denote the best result for each setting. We use serial numbers to represent models, e.g., "①/② + ③" represents query routing from LLaMA-2 or ChatGLM-2 to GPT-3.5-turbo. The number in the parentheses denotes the performance improvement over the best open-source LLM before query routing.

$\mathcal{X} = \left\{ S_i^{(\theta)} = (Q_i, R_i^{(\theta)}) \right\}_{i=1}^{N_{\text{dev}}}$, along with the associated evaluation results $\left\{ C_i^{(\theta)} \right\}_{i=1}^{N_{\text{dev}}}$.

There are two reasonable ways of deriving the label of reference query-response pairs $\mathcal{X}$ in MR. The first is to directly define $C_i^{(\theta)}$ as C_i, i.e. $C_i \triangleq C_i^{(\theta)}$. The second option is to compute the responses from the API-based LLM (Θ), and their evaluation results $\left\{ C_i^{(\Theta)} \right\}_{i=1}^{N_{\text{dev}}}$. We could define the label

$$C_i \triangleq C_i^{(\theta)} - C_i^{(\Theta)}, \tag{5}$$

since $C_i < 0$ denotes the query will be routed and the improvement happens only when $C_i^{(\theta)} < C_i^{(\Theta)}$. We denote the second option by MR$_\Delta$, which considers the actual performance of API-based LLMs.

Accordingly, we could obtain an estimated reliability for $S_i^{(\theta)}$ from the MR approach. Consequently, if the reliability assessment indicates a low accuracy, we can direct Q_i to an API-based large language model (such as GPT-4) to secure a more precise response. Conversely, we preserve the original response for a positive MR result. In an ideal situation, this approach has the potential to enhance performance with moderate costs, as it is generally observed that for the same LLM, a highly accurate response often corresponds to a simple query with less potential for improvement, and vice versa.

4.1.2 Experiment

Settings We leverage reasoning and translation tasks to validate the effectiveness of query routing. We use the same datasets for reasoning tasks as Section 3 and randomly sampled test and development set from the FLORES-200 dataset (NLLB Team et al., 2022). For MR, we prompt Phi-2 in reasoning tasks, and in translation tasks, we adopt `wmt22-cometkiwi-da` (Rei et al., 2022b) for reference-free quality estimation and thus compare the estimated scores between translations. Due to the diverse LLM capabilities and language biases, we have tested two combinations: (1) LLaMA-2 for English and ChatGLM-2 for Chinese tasks, with GPT-3.5-turbo as the API-based model; (2) OpenChat-3.5 for English and Yi for Chinese problems, with GPT-4-turbo as the API. For baselines, we validate the logits ensemble of the open-source models and implement query routing with strategies of entropy-based uncertainty estimation (Han et al., 2024). We omit the result of the latter for OpenChat-3.5 and Yi, since it results in almost no routed queries, probably because the uncertainty threshold is too high to determine false answers.

Effectiveness of Query Routing Guided by Less Capable LLMs In Table 2, we report the average accuracy on reasoning tasks and the average among BLEU (Papineni et al., 2002), BLEURT (Sellam et al., 2020; Pu et al., 2021), and COMET (Rei et al., 2022a) scores on translation tasks. With the query routing approach, we observe significant improvement against single open-source LLMs. Moreover, MR and MR$_\Delta$ manage to gain the highest performance improvement across tasks and

languages with less than half token consumption. Moreover, $\text{MR}_{\triangle}$ consistently outperforms random selection with near half of the token consumption, demonstrating effectiveness of Equation (5).

Relation with Error Detection Performance The performance of our query routing mechanism is closely related to the effectiveness of error detection. For instance, in Table 2, MR outperforms Random and Entropy on absolute performance and token consumption, which denotes MR detects errors in a larger quantity and with higher precision respectively. It suggests that error detection with MR is robust on open-ended generation tasks with continuous labels, e.g., translation.

4.2 Iterative Training Data Refinement

4.2.1 Implementation

Drawing inspiration from studies on data quality for LLM alignment (Zhou et al., 2023), we introduce an iterative training data refinement process. The basic intuition is to continuously filter low-quality data, leading LLMs to concisely learn from more reliable and fewer training samples. Our approach contains two stages (Figure 5 ②): (1) SFT with MR guided data selection and (2) Post-SFT training with both estimated low- and high-quality data from the last training epoch at stage 1.

For the first stage, we extract a small set of queries from the training set and, after each epoch, ask the tuned LLM to respond to those queries. With the generated responses and the queries as references, the MR method could judge whether a single sample in the original dataset matches the quality of the reference. Thus, we could filter data samples that fail the judgment, improving training efficiency and, potentially, LLM performance.

For the second stage, since traditional reinforcement learning (e.g., RLHF (Ouyang et al., 2022)) and contrastive methods (e.g., DPO (Rafailov et al., 2023)) require multiple responses of different human preferences or quality to the same query, they are not compatible. Recently, Ethayarajh et al. (2024) proposed Kahneman-Tversky Optimization (KTO) to align LLMs towards desired and away from undesired query-response pairs collectively. However, we are aware that their objective is still misaligned with our requirement because the low-quality data is derived from the SFT dataset, which is not completely negative. Therefore, in order to incorporate both high- and low-quality data as positive and partially positive samples, we propose positive-KTO (pKTO), whose training objective is

Method	MT-Bench	AlpacaEval 2.0
`Alpaca-13B`	4.53	2.65
`Guanaco-13B`	-	3.47
`Phi-2`	4.52	2.34
`LLaMA-7B`	2.62	0.43
Phi-2-Based Results		
Full	4.42	3.26
Longest	4.56	3.32
Deita	4.33	3.18
Deita (9k)	4.64	3.29
MR (stage 1)	4.70	**3.56**
+ **pKTO** (Stage 2)	**4.77**	3.47
LLaMA-7B-Based Results		
Full	4.36	2.53
Longest	4.18	2.35
Deita	4.37	2.60
Deita (9k)	4.48	2.86
MR (Stage 1)	4.52	2.93
+ **pKTO** (Stage 2)	**4.69**	**3.24**

Table 3: Overall results on iterative training data refinement experiments. The **bold font** denotes best results. "Full" denotes we used the whole Alpaca dataset.

$$\mathcal{L}(\pi_\theta, \pi_{\text{ref}}) \triangleq \mathbb{E}_{(Q,R)\in\mathcal{D}}[\lambda_{\mathbb{I}((Q,R)\in\mathcal{D}_{\text{high}})} \cdot \sigma(v(Q,R))], \quad (6)$$

where

$$z_{\text{ref}} = \mathbb{E}_{(Q',\cdot),(\cdot,R')\in D} \left[\beta\mathcal{L}_{\text{KL}}\left(\pi_\theta(R'|Q')||\pi_{\text{ref}}(R'|Q')\right)\right],$$

$$r(Q,R) = z_{\text{ref}} - \beta\log\frac{\pi_\theta(R|Q)}{\pi_{\text{ref}}(R|Q)},$$

$$v(Q,R) = \begin{cases} r(Q,R) & \text{if } (Q,R)\in\mathcal{D}_{\text{high}} \\ \mathcal{L}_{\text{MSE}}(r(Q,R)) & \text{if } (Q,R)\in\mathcal{D}_{\text{low}} \end{cases},$$

$\mathcal{D} = \{(Q_i, R_i)\}_{i=1}^{N_\mathcal{D}}$ represents the training set, $\mathcal{D}_{\text{high}}, \mathcal{D}_{\text{low}}$ denote the high- and low-quality querie-response pairs from MR results respectively, $\mathcal{L}_{\text{MSE}}$ and $\mathcal{L}_{\text{KL}}$ are the MSE and KL loss respectively, σ is the sigmoid function, π_θ is the trained model, π_{ref} is the reference model which is a copy of untrained π_θ on default, and $\lambda_{\{0,1\}}, \beta$ are hyper-parameters. Intuitively, pKTO differs from KTO only in dealing with low-quality data, where pKTO regulates the reward of these data with MSE loss instead of decreasing it unlimitedly, which prevents LLMs from unlearning the inherent knowledge. We followed the original KTO (Ethayarajh et al., 2024) for implementation. Please refer to Appendix C.2 for comparisons between pKTO, KTO, and DPO.

4.2.2 Experiment

Settings In all experiments, we only use the Alpaca-52k (Taori et al., 2023) dataset with 52,002 samples from `text-davinci-003` (Brown et al., 2020). We

utilize AlpacaEval 2.0 (Li et al., 2023b) and MT-Bench (Zheng et al., 2023a) to benchmark instruction following capabilities. We select strong baselines on SFT data selection, including **Deita** (Liu et al., 2024) and **Longest** (Zhao et al., 2024). Please refer to Appendix B.3 for implementation details. For base models, we choose `Phi-2` and `LLaMA-7B` (Touvron et al., 2023a) for instruction tuning. For Phi-2, our method starts with the original 52k dataset. For LLaMA, Chen et al. (2024) empirically find a 9k subset of Alpaca is the most proper for SFT. We thus adopt the scorer from Deita and extract the top 9k data, noted by Deita (9k), as the initial training set for LLaMA at stage 1.

Effectiveness of Stage 1 We report overall results in Table 3. For each base model, we found iterative training data refinement leads to significant improvement in both benchmarks. In the first stage of MR guided data selection, our method iteratively filters low-quality data after each epoch. After stage 1, Phi-2 and LLaMA already surpass all baselines, indicating MR effectively picks high-quality training samples in a curriculum way (Bengio et al., 2009), which helps align LLMs better compared to select data at the beginning for SFT as baselines.

Effectiveness of Stage 2 We also notice a significant enhancement of stage 2 in Table 3. We report the detailed scores on each turn of LLM responses from MT-Bench in Table 4. From empirical results, pKTO enhances the second-turn communication of LLMs to a great extent while preserving the instruction-following abilities from SFT at stage 1, which is indicated by the slight drops in first-turn scores. In contrast, KTO fails at this setting, which is aligned with our hypothesis in Section 4.2.1. By incorporating the low- and high-quality data distinguished from MR, stage 2 further elicit the capacity of LLMs, especially for multi-turn scenarios.

Method	MT-Bench		
	Turn 1	Turn 2	Avg.
`Alpaca-13B`	4.98	4.09	4.53
`Falcon-40B-instruct`	5.81	4.53	5.17
Phi-2-Based post-SFT Results			
MR (Stage 1)	**6.33**	3.08	4.70
+ KTO (Stage 2)	6.12	2.94	4.53
+ **pKTO** (Stage 2)	6.27	**3.26**	**4.77**
LLaMA-7B-Based post-SFT Results			
MR (Stage 1)	**5.47**	3.58	4.52
+ KTO (Stage 2)	5.15	3.60	4.38
+ **pKTO** (Stage 2)	5.45	**3.93**	**4.69**

Table 4: The multi-turn evaluation results from MT-Bench on iterative training data refinement experiments, which is unfolded from the second column in Table 3. The **bold font** denotes best results for each base model.

5 Related Work

Evaluation of Single LLM Responses Extensive research has been conducted to evaluate responses from LLMs. Studies have focused on estimating uncertainty to gauge the potential reliability of individual answers (Kuhn et al., 2023; Rafailov et al., 2023). Furthermore, LLMs are capable of providing uncertainty scores from itself by fine-tuning (Chen et al., 2023) or black-box techniques (Lin et al., 2023). Additionally, the LLM-as-a-judge approach effectively assesses the accuracy of LLM responses from strong LLMs (Zheng et al., 2023a). Contrarily, our *Meta Ranking* method leverages less capable LLMs and a training-free few-shot judgment by cross-query comparison.

Selective Augmentation on LLMs Recent studies focus on how LLMs can selectively call tools or stronger models only in difficult situations for better efficiency. Tool calls or another trial happens on external feedback from environment (Lin et al., 2023; Shinn et al., 2023). For tasks with explicit criteria, e.g., coding, LLMs can call stronger models after their failure (Zhang et al., 2023). Selection can also be achieved through fine-tuning (Erbacher et al., 2024) or uncertainty estimation (Han et al., 2024). In our approach, MR decides to route queries on complicated open-ended tasks, and we also validate it on coding tasks in Appendix D.3.

Data-Efficient Training for LLMs Coresets (Mirzasoleiman et al., 2020) are used in machine learning for a long period. For LLMs, several data selection methods are developed for SFT (Liu et al., 2024; Zhou et al., 2023; Li et al., 2023b) and post-SFT stages (Gulcehre et al., 2023; Aksitov et al., 2023). Inspired by the iterative training style of the latter, we introduce MR guided iterative training data refinement during SFT.

6 Conclusion

We presents *Meta Ranking* (MR), a novel method effectively enabling less capable LLMs to judge the reliability of single responses. By comparing a target query-response pair with labeled reference pairs, MR outperforms strong baselines in error detection without fine-tuning. Furthermore, the method significantly enhances LLM performance and efficiency in query routing and iterative training data refinement. These findings highlight the potential of MR for broader applications.

Limitations

There are several limitations to our work that we would like to acknowledge:

We have not explored deep in the compatibility between the model training process and the *Meta Ranking* (MR) method. Different model training strategies may affect the effectiveness of MR. It is an interesting direction for future work to study how to better integrate MR with the alignment process (e.g., SFT and post-SFT training) on LLMs.

We have not focused on finding potential applications of the *Meta Ranking* method for strong models. Our current experiments focus on enabling less capable LLMs to judge the reliability of individual responses. It remains an open question what practical usage could incorporate MR with strong models like GPT-4. Exploring the potential applications of MR for strong models is also a direction for future work.

In conclusion, while our proposed *Meta Ranking* method has shown promising results in enabling less capable LLMs to judge the reliability of individual responses and enhancing LLM performance in practical applications, there are still spaces to be explored. We hope that future research can address these limitations and further improve the method.

References

01.AI. 2023. Yi: Building the Next Generation of Open-Source and Bilingual LLMs. https://github.com/01-ai/Yi.

Renat Aksitov, Sobhan Miryoosefi, Zonglin Li, Daliang Li, Sheila Babayan, Kavya Kopparapu, Zachary Fisher, Ruiqi Guo, Sushant Prakash, Pranesh Srinivasan, Manzil Zaheer, Felix Yu, and Sanjiv Kumar. 2023. ReST meets ReAct: Self-Improvement for Multi-Step Reasoning LLM Agent. *Computing Research Repository*, arXiv:2312.10003.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA. Association for Computing Machinery.

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, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 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.

Jiefeng Chen, Jinsung Yoon, Sayna Ebrahimi, Sercan Arik, Tomas Pfister, and Somesh Jha. 2023. Adaptation with self-evaluation to improve selective prediction in LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5190–5213, Singapore. Association for Computational Linguistics.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2024. Alpagasus: Training a Better Alpaca Model with Fewer Data. In *The Twelfth International Conference on Learning Representations*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335, Dublin, Ireland. Association for Computational Linguistics.

Pierre Erbacher, Louis Falissar, Vincent Guigue, and Laure Soulier. 2024. Navigating Uncertainty: Optimizing API Dependency for Hallucination Reduction in Closed-Book Question Answering. *Computing Research Repository*, arXiv:2401.01780.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. KTO: Model Alignment as Prospect Theoretic Optimization. *Computing Research Repository*, arXiv:2402.01306.

Ekaterina Fadeeva, Roman Vashurin, Akim Tsvigun, Artem Vazhentsev, Sergey Petrakov, Kirill Fedyanin, Daniil Vasilev, Elizaveta Goncharova, Alexander Panchenko, Maxim Panov, Timothy Baldwin, and Artem Shelmanov. 2023. LM-polygraph: Uncertainty estimation for language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 446–461, Singapore. Association for Computational Linguistics.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. 2023. Reinforced Self-Training (ReST) for Language Modeling. *Computing Research Repository*, arXiv:2308.08998. Version 2.

Jiuzhou Han, Wray Buntine, and Ehsan Shareghi. 2024. Towards Uncertainty-Aware Language Agent. *Computing Research Repository*, arXiv:2401.14016. Version 2.

Shibo Hao, Tianyang Liu, Zhen Wang, and Zhiting Hu. 2023. ToolkenGPT: Augmenting Frozen Language Models with Massive Tools via Tool Embeddings. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021a. Aligning {AI} With Shared Human Values. In *International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021b. Measuring Massive Multitask Language Understanding. In *Proceedings of the 9th International Conference on Learning Representations*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. *Computing Research Repository*, arXiv:2311.05232.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. 2022. Language Models (Mostly) Know What They Know. *Computing Research Repository*, arXiv:2207.05221.

Pei Ke, Bosi Wen, Zhuoer Feng, Xiao Liu, Xuanyu Lei, Jiale Cheng, Shengyuan Wang, Aohan Zeng, Yuxiao Dong, Hongning Wang, Jie Tang, and Minlie Huang. 2023. CritiqueLLM: Scaling LLM-as-Critic for Effective and Explainable Evaluation of Large Language Model Generation. *Computing Research Repository*, arXiv:2311.18702.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation. In *The Eleventh International Conference on Learning Representations*.

Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. 2022. JGLUE: Japanese general language understanding evaluation. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2957–2966, Marseille, France. European Language Resources Association.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. 2023a. CMMLU: Measuring massive multitask language understanding in Chinese. *Computing Research Repository*, arXiv:2306.09212.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023b. AlpacaEval: An Automatic Evaluator of Instruction-following Models. `https://github.com/tatsu-lab/alpaca_eval`.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023a. Textbooks Are All You Need II: phi-1.5 technical report. *Computing Research Repository*, arXiv:2309.05463.

Yunshui Li, Binyuan Hui, Xiaobo Xia, Jiaxi Yang, Min Yang, Lei Zhang, Shuzheng Si, Junhao Liu, Tongliang Liu, Fei Huang, and Yongbin Li. 2023b. One Shot Learning as Instruction Data Prospector for Large Language Models. *Computing Research Repository*, arXiv:2312.10302.

Bill Yuchen Lin, Yicheng Fu, Karina Yang, Faeze Brahman, Shiyu Huang, Chandra Bhagavatula, Prithviraj Ammanabrolu, Yejin Choi, and Xiang Ren. 2023. SwiftSage: A Generative Agent with Fast and Slow Thinking for Complex Interactive Tasks. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2023. Generating with Confidence: Uncertainty Quantification for Black-box Large Language Models. *Computing Research Repository*, arXiv:2305.19187. Version 2.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2024. What Makes Good Data for Alignment? A Comprehensive Study of Automatic Data Selection in Instruction Tuning. In *The Twelfth International Conference on Learning Representations*.

Andrey Malinin and Mark Gales. 2021. Uncertainty Estimation in Autoregressive Structured Prediction. In *International Conference on Learning Representations*.

Sabrina J. Mielke, Arthur Szlam, Emily Dinan, and Y-Lan Boureau. 2022. Reducing conversational agents' overconfidence through linguistic calibration. *Transactions of the Association for Computational Linguistics*, 10:857–872.

Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. 2020. Coresets for data-efficient training of machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6950–6960. PMLR.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia-Gonzalez, Prangthip Hansanti,

John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No Language Left Behind: Scaling Human-Centered Machine Translation. *Computing Research Repository*, arXiv:1902.01382. Version 3.

OpenAI. 2023. GPT-4 Technical Report. *Computing Research Repository*, arXiv:2303.08774. Version 4.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

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.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Amy Pu, Hyung Won Chung, Ankur Parikh, Sebastian Gehrmann, and Thibault Sellam. 2021. Learning compact metrics for MT. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 751–762, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Ricardo Rei, José G. C. de Souza, Duarte Alves, Chrysoula Zerva, Ana C Farinha, Taisiya Glushkova, Alon Lavie, Luisa Coheur, and André F. T. Martins. 2022a. COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Ricardo Rei, Marcos Treviso, Nuno M. Guerreiro, Chrysoula Zerva, Ana C Farinha, Christine Maroti, José G. C. de Souza, Taisiya Glushkova, Duarte Alves, Luisa Coheur, Alon Lavie, and André F. T. Martins. 2022b. CometKiwi: IST-unbabel 2022 submission for the quality estimation shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 634–645, Abu Dhabi, United Arab Emirates (Hybrid). 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.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford Alpaca: An Instruction-following LLaMA model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. LLaMA: Open and Efficient Foundation Language Models. *Computing Research Repository*, arXiv:2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open Foundation and Fine-Tuned Chat Models. *Computing Research Repository*, arXiv:2307.09288. Version 2.

Guan Wang, Sijie Cheng, Xianyuan Zhan, Xiangang Li, Sen Song, and Yang Liu. 2024a. OpenChat: Advancing Open-source Language Models with Mixed-Quality Data. In *The Twelfth International Conference on Learning Representations*.

Yidong Wang, Zhuohao Yu, Zhengran Zeng, Linyi Yang, Wenjin Yao, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, Wei Ye, Shikun Zhang, and Yue Zhang. 2024b. PandaLM: An Automatic Evaluation Benchmark for LLM Instruction Tuning Optimization. In *The Twelfth International Conference on Learning Representations*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. 2024. Hallucination is Inevitable: An Innate Limitation of Large Language Models. *Computing Research Repository*, arXiv:2401.11817.

Yiming Yan, Tao Wang, Chengqi Zhao, Shujian Huang, Jiajun Chen, and Mingxuan Wang. 2023. BLEURT has universal translations: An analysis of automatic metrics by minimum risk training. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5428–5443, Toronto, Canada. Association for Computational Linguistics.

Yuqing Yang, Ethan Chern, Xipeng Qiu, Graham Neubig, and Pengfei Liu. 2023. Alignment for Honesty. *Computing Research Repository*, arXiv:2312.07000.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Jiawen Wu, Xipeng Qiu, and Xuanjing Huang. 2023. Do large language models know what they don't know? In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8653–8665, Toronto, Canada. Association for Computational Linguistics.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. 2023. GLM-130B: An Open Bilingual Pre-trained Model. In *The Eleventh International Conference on Learning Representations*.

Jieyu Zhang, Ranjay Krishna, Ahmed H. Awadallah, and Chi Wang. 2023. EcoAssistant: Using LLM Assistant More Affordably and Accurately. *Computing Research Repository*, arXiv:2310.03046.

Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. 2024. Long Is More for Alignment: A Simple but Tough-to-Beat Baseline for Instruction Fine-Tuning. *Computing Research Repository*, arXiv:2402.04833.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023a. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Qinkai Zheng, Xiao Xia, Xu Zou, Yuxiao Dong, Shan Wang, Yufei Xue, Zihan Wang, Lei Shen, Andi Wang, Yang Li, Teng Su, Zhilin Yang, and Jie Tang. 2023b. CodeGeeX: A Pre-Trained Model for Code Generation with Multilingual Evaluations on HumanEval-X. *Computing Research Repository*, arXiv:2303.17568.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, LILI YU, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. LIMA: Less Is More for Alignment. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. 2023a. JudgeLM: Fine-tuned Large Language Models are Scalable Judges. *Computing Research Repository*, arXiv:2310.17631.

Wenhao Zhu, Yunzhe Lv, Qingxiu Dong, Fei Yuan, Jingjing Xu, Shujian Huang, Lingpeng Kong, Jiajun Chen, and Lei Li. 2023b. Extrapolating Large Language Models to Non-English by Aligning Languages. *Computing Research Repository*, arXiv:2308.04948. Version 2.

A Broader Impact

The *Meta Ranking* (MR) method presented in this paper has the potential to significantly influence the field of LLMs and their applications. Here, we discuss the broader impact of our work in several key areas:

Data and LLM Response Curation: Meta Ranking enables the use of smaller, less resource-intensive models for single response judgement, which previously requires large, expensive models. Thus, MR is more cost-efficient and practical for landing in real-world scenarios. Also, MR does not inherently incorporate risks in the methodology, while the language model MR utilizes could contain potential risks from its pretrain data. The application of MR, including query routing and iterative training data refinement, may result in risky results due to the nature of the application. However, MR could actually be used in risk mitigation for LLMs, by identifying and filtering LLM responses with potential risks.

LLM Inference and Training: The MR method can improve the efficiency and effectiveness of LLM inference and training. By routing queries to the most appropriate LLMs based on reliability assessments, MR can save computational resources and improve response times, making LLMs more practical for real-world applications. Additionally,

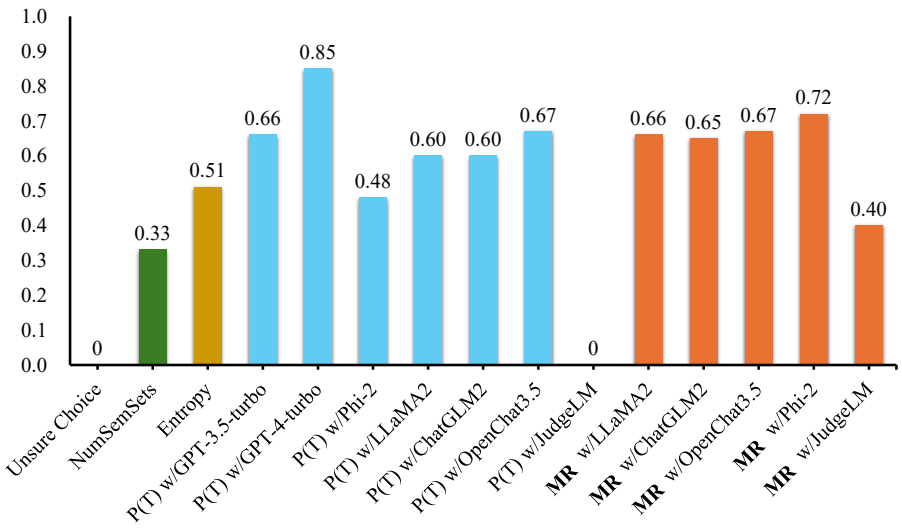

Figure 6: The F1 score of all methods on error detection experiments for responses from LLaMA-2 on the MMLU dataset.

the iterative training data refinement enabled by MR can lead to more accurate and reliable LLMs, which is crucial for maintaining public trust in AI systems.

In conclusion, the Meta Ranking method not only enhances the capabilities of less capable LLMs but also has the potential to transform how we develop, deploy, and interact with AI systems, leading to a more reliable, efficient, and equitable integration of AI in various aspects of our lives.

Algorithm 1: Meta Ranking

Input : Target query-response pair $S_{\text{target}} = (Q_{\text{target}}, R_{\text{target}})$, reference query-response pairs $\mathcal{X} = \{S_i = (Q_i, R_i)\}_{i=1}^{N}$, and the reliability label C_i for each S_i, hyperparameters δ_{+1}, δ_0, and δ_{-1}

Output : A boolean indicator $\mathcal{I}$ of the reliability of the target response

$s \leftarrow 0$;
for $i \leftarrow 1$ **to** N **do**
  $r \leftarrow \text{MR}\,(S_{\text{target}}, S_i)$;
  $r \leftarrow \text{sgn}(C_i) \times r$;
  $s \leftarrow s + C_i \times \delta_r$;
end
if $s \geq 0$ **then** $\mathcal{I} \leftarrow$ True **else** $\mathcal{I} \leftarrow$ False;

B Implementation Details

We demonstrate the detail process of *Meta Ranking* in Algorithm 1. Under this core technique, we elaborate implementation details of experiments below.

For general settings, all experiments in this paper were conducted on two types of servers: 8*A800 and 8*V100. The A800 server is equipped with 8 NVIDIA `A800-SXM4-80GB` GPUs. The V100 server features 8 NVIDIA `Tesla V100-PCIE-32GB` GPUs. For open-source LLMs, we use Phi-2 to denote `Phi-2-2.7B`[1] (Li et al., 2023a), LLaMA to denote `LLaMA-7B`[2] (Touvron et al., 2023a), LLaMA-2 to denote `LLaMA-2-chat-7B`[3] (Touvron et al., 2023b), ChatGLM-2 for `ChatGLM-2-6B`[4] (Zeng et al., 2023), OpenChat-3.5 for `OpenChat-3.5-7B`[5] (Wang et al., 2024a), Yi for `Yi-6B-Chat`[6] (01.AI, 2023), and GPT-3.5-turbo and GPT-4-turbo for `GPT-3.5-turbo-1106` and `GPT-4-1106-preview` (OpenAI, 2023). For reproduction, we set the `temperature` to 0 for LLM generation without specification. And the maximum tokens for LLM generation is set to 512.

For clarification, we have ensured that the use of pretrained and instruction-tuned LLMs and datasets is consistent with their intended use and licenses. Furthermore, the derivatives of these data and instruction-tuned models based on MR should not be used in consideration of the original access con-

[1] https://huggingface.co/microsoft/phi-2
[2] https://huggingface.co/huggyllama/llama-7b
[3] https://huggingface.co/meta-llama/Llama-2-7b-chat
[4] https://huggingface.co/THUDM/chatglm2-6b
[5] https://huggingface.co/openchat/openchat_3.5
[6] https://huggingface.co/01-ai/Yi-6B-Chat

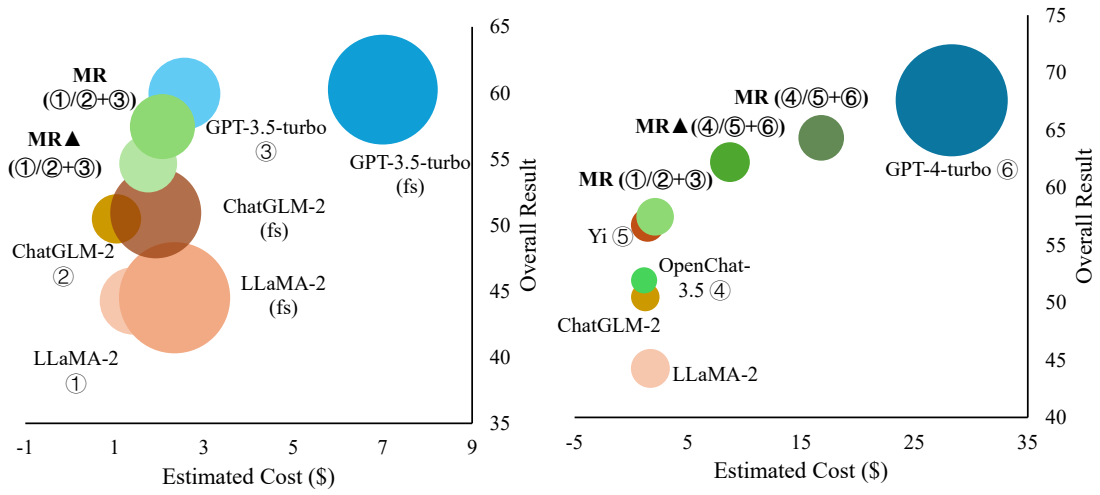

Figure 7: The overall results and estimated costs in query routing experiments. Additionally, we use the width of circles to illustrate the latency of inference (Sec./Iter.) relatively. We devide the two settings into subfigures. "(fs)" denotes few-shot results.

ditions and ethical guidelines.[7]

B.1 Error Detection

Detailed Settings For the MMLU dataset (Hendrycks et al., 2021b), we evenly selected subjects in each category, including "Abstract Algebra" and "College Mathematics" for STEM, "Prehistory" and "Moral Scenarios" for humanities, "Econometrics" and "Professional Psychology" for social sciences, and "Global Facts" and "Professional Accounting" for others. For the CMMLU dataset (Li et al., 2023a), we select "College Actuarial Science" for STEM, "World History" for Humanities, "Security Study" for social sciences, "Traditional Chinese Medicine" for China-specific subjects, and "Human Sexuality" for others. We assign the hyper-parameter in MR as following: $\delta_{+1} = 1, \delta_0 = 1, \delta_{-1} = -0.5$ for MMLU, and $\delta_{+1} = 1, \delta_0 = 0.5, \delta_{-1} = -0.25$ for CMMLU. We calculate the accuracy by exactly matching the generated choice (e.g., A, B, C, or D) with the ground truth. We use the five examples in the development set along with generated responses for the reference query-response pair in MR. According to Algorithm 1, MR inherently has no positional bias over reference pairs.

Baseline Implementations For (1) **Unsure Choice**, we include an additional option, "`(E) Not Sure`", allowing the LLM to admit uncertainty in a zero-shot manner on questions it might answer incorrectly. For uncertainty measurement (2) **NumSemSets** (Lin et al., 2023), given that the responses of different choices inherently form semantic sets (Kuhn et al., 2023), we sample five times on each question with the same LLM with the `temperature` of 0.8, and decide on an incorrect answer if the number of semantic sets is larger than all correct examples. (3) **Entropy** (Han et al., 2024) measures uncertainty of LLM responses in a white-box way. A response is decided wrong when its entropy is lower than all correct examples, following the calibration method in Han et al. (2024). Lastly, (4) **P(T)**: the P(True) (Kadavath et al., 2022) baseline directly asks an LLM about the correctness of a query-response pair, thus able to be implemented on different LLMs, including the aforementioned open-source LLMs and API-based `GPT-3.5-turbo-1106` and `GPT-4-1106-preview`. We implement Entropy and P(T) in a few-shot manner, with examples in the development set. We shuffled the few-shot examples to eliminate the positional bias. For the one-shot experiments in Figure 4, we use the first sample in the development set.

Biases in Error Detection To validate the potential biases in the error detection experiments (e.g., judging all responses as false ones) in Figure 3, we report the micro F1 scores under the identical setting in Figure 6. We can observe that the F1 results approximately follow the trend of precision scores in Figure 3, and for MR methods, the F1 scores are closed to or surpass GPT-3.5-turbo, demonstrating unbiased judgement for correct and incorrect responses.

[7]We will release our code once the paper is accepted.

Model	Zh-En			En-Zh			#Token (Local)	#Token (API)
	BLEU	BLEURT	COMET	BLEU	BLEURT	COMET	($\times 10^4$)	($\times 10^4$)
Phi-2	8.1	49.60	69.27	2.5	23.28	43.77	49.82	
LLaMA-2 (①)	20.1	71.13	84.35	22.7	55.30	75.99	17.52	
ChatGLM-2 (②)	20.8	70.52	83.97	36.3	68.21	85.40	7.43	-
OpenChat-3.5 (④)	24.4	73..85	85.80	32.1	66.24	83.97	8.92	
Yi (⑤)	23.8	73.39	85.41	31.1	69.16	85.94	6.78	
GPT-3.5-turbo (③)	27.8	76.04	87.16	45.7	73.12	88.59	-	8.72
GPT-4-turbo (⑥)	29.6	77.04	87.55	46.9	73.86	89.08		8.78
MR (①/② + ③)	23.7	74.17	86.20	41.6	71.70	88.00	8.28	4.01
MR$_\triangle$ (①/② + ③)	21.3	72.70	86.20	29.5	70.77	87.35	8.28	2.26
MR (④/⑤ + ⑥)	25.5	75.52	86.79	43.3	72.66	88.38	7.86	3.22
MR$_\triangle$ (④/⑤ + ⑥)	24.4	74.30	86.11	432.6	72.19	88.05	7.86	2.07

Table 5: Detailed results on translation tasks in Table 2. "**#Token** (Local)" denotes the total number of prompt and generated tokens during inference of open-source LLMs.

B.2 Query Routing

Detailed Settings For the translation task, we randomly extracted 400 parallel sentences from the dev-test set as the test set and 20 sentences from the development set respectively, in Chinese and English from Flores-200 (NLLB Team et al., 2022). For evaluation, we adopt SacreBLEU (Post, 2018) for BLEU calculation[8], BLEURT-20 (Pu et al., 2021) for BLEURT scores (Yan et al., 2023), and `wmt22-comet-da`[9] for COMET scores (Rei et al., 2022a). We report the detailed results of translation tasks in Table 5. For the logit ensemble baseline, we map the vocabulary from one LLM to another, thus we can add logits from different LLM with an equal magnitude. We adopt the manner from Hao et al. (2023) to train a single token for LLMs to identify the language of generation from the multilingual Alpaca dataset released by Zhu et al. (2023b). Thus, LLMs can automatically switch LLMs for Chinese and English tasks, resulting in the combinations of LLaMA-2 and ChatGLM-2 as well as OpenChat-3.5 and Yi as a complete system, as depicted in Section 4.1. Thus, we view the results of query routing in the same row of Table 2 as a whole. We follow the same implementation of MR from error detection experiments (Appendix B.1) on reasoning tasks. For translation tasks, we set $\delta_{+1} = 1, \delta_0 = 0, \delta_{-1} = -1$ in Algorithm 1. We implement the cross-query comparison with the reference-free quality estimation model `wmt22-cometkiwi-da`[10] (Rei et al., 2022b).

[8]The signature is "nrefs:1+case:mixed+eff:no+smooth:exp +version:2.3.1".

[9]https://huggingface.co/Unbabel/wmt22-comet-da

[10]https://huggingface.co/Unbabel/wmt22-cometkiwi-da

Cost Estimation In query routing experiments (under the same setting of Table 2), we estimate the cost of each model and method in Figure 7 in US dollars, with the reference of the pricing of AWS cloud servers, from which we estimate the cost of local running LLMs, and the OpenAI pricing on GPT-3.5-turbo and GPT-4-turbo.[11] We also measure the average inference time for each sample for each method, and the network latency is contained for API-based GPT-3.5-turbo and GPT-4-turbo. Empirically, we demonstrate that query routing with MR achieves comparable performance to API-based LLMs with moderate costs on real money.

B.3 Iterative Training Data Refinement

Detailed Settings For dataset, we manually reexamine the anonymity of Alpaca 52k (Taori et al., 2023). We use the default setting of the MT-Bench (Zheng et al., 2023a) and AlpacaEval 2.0 (Li et al., 2023b) benchmarks. Since AlpacaEval 2.0 uses a non-zero `temperature` for evaluation, thus we run the evaluation for three times and report the median value. We follows the original hyper-parameter setting as (Taori et al., 2023) for all baselines and our method at stage 1, except we use a batch size of 128 for fine-tuning Phi-2 and of 256 for fine-tuning LLaMA. The number of training epochs is 3 for all baselines and our method at stage 1. Our method at stage 2 uses the same hyper-parameters as KTO (Ethayarajh et al., 2024). For baselines, Longest (Zhao et al., 2024) selects 1k samples with the longest responses; Deita (Liu et al., 2024) provides distilled scorers from GPT-3.5 for scoring each training sample and we ex-

[11]AWS server pricing and OpenAI pricing URLs.

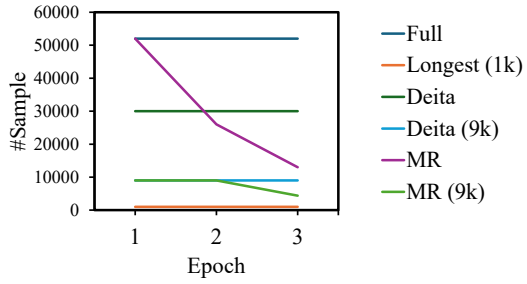

Figure 8: Training Samples of different methods on the Alpaca-52k dataset in the SFT process.

Method	#Token ($\times 10^5$)
Full	132.93
Longest	10.08
Deita	96.09
Deita (9k)	39.81
MR	87.14
MR (9k)	33.19

Table 6: The overall number of training tokens in the SFT and post-SFT training process of all methods, which is calculated by LLaMA-2 Tokenizer.

tract 30k samples with highest scores. For Deita (9k), we apply the same scorers for 9k samples. Specifically, we duplicate the training data at the third epoch for Phi-2 on our method at stage 1, due to compatibility issue with the cosine learning rate scheduler. For MR implementation, we set $\delta_{+1} = 1, \delta_0 = 0, \delta_{-1} = -1$ in Algorithm 1.

Training Tokens Comparison Results We demonstrate the training samples in Figure 8 and the training tokens in Table 6. MR denotes our method at stage 1 and "Full" denotes the whole Alpaca dataset. We exhibit that our method achieves superior performance in Table 3 with a moderate amount of training samples and tokens. For clarity, `Alpaca-13B` in Table 3 and Table 4 is LLaMA-13B fine-tuned on the whole Alpaca dataset.

C Mathematical Arguments and Steps

C.1 Explanation on *Meta Ranking* Methodology

In this section, we provide the proof for

$$\operatorname{sgn}(\Delta s_i) = \operatorname{sgn}(C_{\text{target}} - C_i), \quad \text{if } C_{\text{target}} - C_i \neq 0, C_i \neq 0, \tag{7}$$

and theoretically explain why the *Meta Ranking* process could approximately determine the reliability of the response a query-response pair, according to Section 2. From Equation (3), we note that $\operatorname{sgn}(\Delta s_i) = \operatorname{sgn}(C_i) \cdot \operatorname{sgn}(\delta_{\operatorname{sgn}(C_i)\cdot r_i})$, when $C_i \neq 0$.

Recall that $r_i \in \{\pm 1, 0\}$, which represents the MR results of the reliability of the target query-response pair $(Q_{\text{target}}, R_{\text{target}})$ with another pair (Q_i, R_i). Therefore, $\operatorname{sgn}(r_i) = \operatorname{sgn}(C_{\text{target}} - C_i) \neq 0$ is valid when MR results are correct.

Note that we have set hyper-parameters $\delta_{+1} > 0 > \delta_{-1}$, indicating $\operatorname{sgn}(\delta_{\operatorname{sgn}(C_i)\cdot r_i}) = \operatorname{sgn}(C_i) \cdot \operatorname{sgn}(r_i)$, when $C_i \neq 0$. Given the values of the sgn function, we notice that

$$\begin{aligned} \operatorname{sgn}(\delta_{\operatorname{sgn}(C_i)\cdot r_i}) &= \operatorname{sgn}(r_i) \cdot \operatorname{sgn}(C_i) \\ &= \operatorname{sgn}(r_i)/\operatorname{sgn}(C_i) \\ &= \operatorname{sgn}(C_{\text{target}} - C_i)/\operatorname{sgn}(C_i). \end{aligned} \tag{8}$$

Identically, we arrive at

$$\begin{aligned} \operatorname{sgn}(\Delta s_i) &= \operatorname{sgn}(C_i) \cdot \operatorname{sgn}(\delta_{\operatorname{sgn}(C_i)\cdot r_i}) \\ &= \operatorname{sgn}(C_{\text{target}} - C_i). \end{aligned} \tag{9}$$

When $C_i = 0$, in case of correctness, it indicates the i-th query-response pair stands neutral. Intuitively, it is hard to tell the correctness of the target pair based on whatever MR results, due to the lack of understandings on what correctness is. This also matches the formulation of Equation (3).

Furthermore, consider Equation (4). Given

$$C_{\text{target}} - C_{\text{avg}} = \frac{1}{N} \sum_{i=1}^{N} (C_{\text{target}} - C_i) \tag{10}$$

and Equation (7), we can similarly view $\operatorname{sgn}(s)$ as an approximation of the sign of $C_{\text{target}} - C_{\text{avg}}$ by viewing $\operatorname{sgn}(\Delta s_i)$ as the approximation of $\operatorname{sgn}(C_{\text{target}} - C_i)$.

In summary, the sign agreement ensures that the expression $C_{\text{target}} - C_{\text{avg}}$ is legitimately approximated based on the MR method.

C.2 Difference on the objective of pKTO, KTO, and DPO

DPO (Rafailov et al., 2023) and KTO (Ethayarajh et al., 2024) are shown to be effecive on post-SFT training We rewrite the objective of positive-KTO (pKTO) in Section 4.2.1:

$$\mathcal{L}_{\text{pKTO}}(\pi_\theta, \pi_{\text{ref}}) = \mathbb{E}_{(Q,R)\in\mathcal{D}}[\lambda_{\mathbb{I}((Q,R)\in\mathcal{D}_{\text{high}})} \cdot \sigma(v(Q,R))],$$

where

$$z_{\text{ref}} = \mathbb{E}_{(Q',\cdot),(\cdot,R')\in D} [\beta \mathcal{L}_{\text{KL}}(\pi_\theta(R'|Q')||\pi_{\text{ref}}(R'|Q'))],$$

$$r(Q,R) = z_{\text{ref}} - \beta \log \frac{\pi_\theta(R|Q)}{\pi_{\text{ref}}(R|Q)},$$

$$v(Q,R) = \begin{cases} r(Q,R) & \text{if } (Q,R) \in \mathcal{D}_{\text{high}} \\ \mathcal{L}_{\text{MSE}}(r(Q,R)) & \text{if } (Q,R) \in \mathcal{D}_{\text{low}} \end{cases}.$$

In our formulation, we can rewrite the objective of KTO (Ethayarajh et al., 2024) by

$$\mathcal{L}_{\text{KTO}}(\pi_\theta, \pi_{\text{ref}}) = \mathbb{E}_{(Q,R)\in\mathcal{D}} \lambda_{\mathbb{I}((Q,R)\in\mathcal{D}_{\text{high}}} \cdot \sigma(v_{\text{KTO}}(Q,R)), \quad (11)$$

where

$$v_{\text{KTO}}(Q,R) = \begin{cases} r(Q,R) & \text{if } (Q,R) \in \mathcal{D}_{\text{high}} \\ -r(Q,R) & \text{if } (Q,R) \in \mathcal{D}_{\text{low}} \end{cases}.$$

Thus, the main difference between pKTO and KTO is the handling of low-quality data for LLM training. In pKTO, a mean squared error (MSE) loss is applied to the reward function $r(Q,R)$ for data samples in $\mathcal{D}_{\text{low}}$, which aims to limit the variation of discrepancy between the predicted and reference policies in $\log \frac{\pi_\theta(R|Q)}{\pi_{\text{ref}}(R|Q)}$. This encourages the policy to improve its performance without unlearning improtant knowledge within D_{low}. On the other hand, KTO simply takes the negative of $r(Q,R)$ for undesired data samples, driving the policy away. This difference in the treatment of low-quality regions leads to distinct optimization behaviors and can impact the overall performance and suitable scenarios, which aligns to the experimental results in Table 3.

The Direct Preference Optimization (DPO) approach, which is another variant in this domain, can also be contrasted with pKTO and KTO. DPO modifies the objective to focus on both policy improvement and preference learning, which can be written as:

$$\mathcal{L}_{\text{DPO}}(\pi\theta, \pi_{\text{ref}}) = \mathbb{E}_{(Q,R)\in\mathcal{D}} [-\log\sigma(\beta \log \frac{\pi_\theta(R_{\text{bad}|Q})}{\pi_{\text{ref}}(R_{\text{bad}|Q})} - \beta \log \frac{\pi_\theta(R_{\text{good}|Q})}{\pi_{\text{ref}}(R_{\text{good}|Q})})], \quad (12)$$

where R_{bad} and R_{good} denote a relatively good and bad response pair to the same query Q, in terms of correctness, human preferences, etc. This definition limits its generalization to incorporate queries with single responses.

In summary, while pKTO, KTO, and DPO share similarities in their overall structure, their distinct treatments of reward functions and their requirements of data set them apart, leading to different trade-offs in policy optimization in different scenarios.

D Additional Experiments

D.1 Error Detection on Japanese reasoning tasks

Method	Precision	F1
P(T) w/ OpenChat-3.5	0.03	0.07
MR w/ Phi-2	0.34	0.51
MR w/ Phi-2 (fine-tuned)	0.55	0.63

Table 7: Results on error detection experiments on Japanese reasoning tasks.

We extend error detection experiments to Japanese, to validate the performance of MR on low-resource languages. We use JCommonsenseQA dataset (Kurihara et al., 2022) and OpenChat for response generation, which have reached an accuracy of 0.74. We report the precision and F1 score of MR with Phi-2 in Table 7, with best performing methods with open-source LLMs in Figure 3. We observe that the performance of P(T) with OpenChat-3.5 perform poorly, denoting low-resource languages do have a negative influence for LLM judgement. Though, MR with Phi-2 faces performance drops to a smaller extent compared to results in Table 1, but still fails to outperform random selection.

We then utilize the training set of JCommonsenseQA and MMLU (Hendrycks et al., 2021b) to fine-tune Phi-2 on cross-query comparisons, with the prompt template in Appendix F, denoted as "Phi-2 (fine-tuned)" in Table 7. We follow the same setting for fine-tuning as Appendix B.3. We found that the fine-tuning does greatly influence the performance of MR on low-resource languages. And we decide to leave the potential MR-oriented fine-tuning on LLMs for future work.

D.2 Query Routing with Fine-tuned LLMs

As summarized by Fadeeva et al. (2023), there are a few training-based methods for uncertainty

Model	Routing Strategy	Pass@1	Pass@10	Pass@100	#Token (API) (Relative Value)
LLaMA-2	-	21.12	34.56	44.35	-
OpenChat-3.5 (①)		39.15	56.54	60.63	
ChatGLM-2		9.02	17.14	26.53	
Yi		19.45	35.83	46.05	
GPT-3.5-turbo (②)		<u>52.12</u>	<u>59.43</u>	<u>62.01</u>	1.00
① + ②	**MR** w/ output matching	**53.23**	**61.37**	**62.80**	**0.56**

Table 8: The result of query routing on the code translation (Python-Java) task. The **bold font** denotes the best result using query routing and the <u>underlined numbers</u> denote the best result for each setting.

Model	Routing Strategy	Accuracy	#Token (API)
LLaMA-2-chat-7B (①)	-	34.37	-
ASPIRE (②)		34.37	
LLaMA-2-chat-13B		35.12	
confucius-multisample (③)		34.05	
GPT-3.5-turbo-1106 (④)		<u>52.91</u>	1.00
② + ④	Uncertainty	42.30	0.25
③ + ④	Direct	36.05	0.12
① + ④	**MR**	**48.76**	0.38

Table 9: The result of query routing on different fine-tuning-based methods on the MMLU dataset. "**#Token** (API)" denote the GPT-3.5-turbo token consumption in relative values. The **bold font** denotes the best result using query routing and the <u>underlined numbers</u> denote the best result for each setting.

estimation (Malinin and Gales, 2021), which can be utilized in query routing. Following the ASPIRE framework (Chen et al., 2023) which leverages parameter-efficient training, we tuned LLaMA-2-chat-7B and the calibration process proposed by Han et al. (2024), that a response is considered incorrect when its uncertainty value is lower than all correct examples. There is also another training technique named alignment for honesty (Yang et al., 2023), which trains LLMs towards acknowledgment of their unknown queries. We tested a honesty-aligned model titled "Confucius" based on LLaMA-2-chat-13B.[12] We route the query to GPT-3.5 when the model outputs that it does not know the answer, which we termed "direct" strategy for query routing.

The results on the MMLU dataset is shown in Table 9, where fine-tuning-based methods provide only marginal improvement in the query routing experiment.

[12]The "Confucius" model is released on GAIR/*confucius-multisample* and under Llama 2 license.

D.3 Query Routing on Code Translation Tasks

We adopted the approach proposed by Codegeex (Zheng et al., 2023b) to assess performance on its HumanEval-X dataset. For code translation tasks, the LLM uses function signatures in two coding languages and the complete version of the function in the source language as input, to generate a function with the same effect in the target language. In our method, we utilized the example arguments in the function signature to feed into the function in the source language and from generation, yielding a twin of outputs. Subsequently, we can compare the outputs by exactly matching, which serves as a explicit criteria for correctness judgement. In MR (Algorithm 1), the label of a query-response pair is defined as the match rate of generated and source functions in this case.

The result from Python to Java is shown in Table 8. We use "pass@n" to denote the correctness of translation of sampling for n times. Please refer to Appendix F for prompt details. The result of MR has surpassed GPt-3.5-turbo, by routing queries from OpenChat-3.5 to GPT-3.5-turbo, indicating the effectiveness of MR on tasks with explicit criteria (e.g., exactly matching for function outputs).

E Case Study

E.1 Query Routing Trajectories

Please refer to Table 10, Table 11, and Table 12 for cases of query routing trajectories.

E.2 Refined Training Data

We sampled three data points from filtered training data and the data after refinement in the iterative training data refinement experiment. We exhibit them in Table 13.

F Prompt Templates

We demonstrate all prompt templates we used in error detection, query routing, and iterative training data refinement tasks in Table 14. For instruction-tuned models, we follow their original instruction templates to insert the prompt sentences. For reference examples, we directly append the examples in the same format as incoming the query and response, and concatenate them as a suffix for language models.

Query (MMLU)	Which of the following are alternative names for the dependent variable (usually denoted by y) in linear regression analysis? (i) The regressand (ii) The regressor (iii) The explained variable (iv) The explanatory variable: A) (ii) and (iv) only, B) (i) and (iii) only, C) (i), (ii), and (iii) only, D) (i), (ii), (iii), and (iv)
Response (OpenChat-3.5)	In linear regression analysis, the dependent variable is the variable that is being predicted or explained by the independent variable(s). The dependent variable is usually denoted by y. (i) The regressand: This term refers to the dependent variable in a regression analysis. It is the variable that is being predicted or explained by the independent variable(s). (ii) The regressor: This term refers to the independent variable(s) in a regression analysis. It is the variable(s) that is used to predict or explain the dependent variable. (iii) The explained variable: This term also refers to the dependent variable in a regression analysis. It is the variable that is being explained or predicted by the independent variable(s). (iv) The explanatory variable: This term refers to the independent variable(s) in a regression analysis. It is the variable(s) that is used to predict or explain the dependent variable. Based on the definitions above, the correct answer is (i), (ii), and (iii) only. Therefore, the answer is **C**. *(False)*
Response (GPT-4-turbo)	In linear regression analysis, the dependent variable is the variable that is being predicted or explained. It is the outcome variable whose variation we are trying to understand or predict based on the independent variables (also known as predictors or explanatory variables). Here are the correct associations for the terms given: (i) The regressand - This is another term for the dependent variable in regression analysis. (ii) The regressor - This term refers to the independent variable(s) in regression analysis, not the dependent variable. (iii) The explained variable - This is another term for the dependent variable, as it is the variable whose variation is being explained by the independent variables. (iv) The explanatory variable - This term refers to the independent variable(s) that are used to explain the variation in the dependent variable. Therefore, the correct alternative names for the dependent variable in linear regression analysis are (i) the regressand and (iii) the explained variable. The answer is: B) (i) and (iii) only (**B**). *(True)*

Table 10: A case in query routing experiments on the MMLU dataset (Section 4.1). The query routes to GPT-4-turbo and gets corrected.

Query (MMLU)	For which of these two scenarios does the main character (who uses I/me/my) do something clearly morally wrong, according to ordinary moral standards in the US as of 2020? Scenario 1 | I said my stomach hurt because I felt like leaving work early. Scenario 2 | As my parents were sleeping, i locked their gun safe.: A) Wrong, Wrong, B) Wrong, Not wrong, C) Not wrong, Wrong, D) Not wrong, Not wrong
Response (LLaMA-2)	**B**. Wrong, Not wrong Explanation: In Scenario 1, the main character (who uses I/me/my) says they have a stomachache and want to leave work early. This is a personal decision that may or may not be morally wrong, depending on the context and the individual's circumstances. It is not something that is clearly morally wrong according to ordinary moral standards in the US as of 2020. In Scenario 2, the main character (who uses I/me/my) locks their parents' gun safe while they are sleeping. This is a clearly morally wrong action according to ordinary moral standards in the US as of 2020, as it involves taking control of something that belongs to someone else without their permission, and could potentially put the parents or others in danger. *(True)*
Response (GPT-3.5-turbo)	(N/A)

Table 11: A case in query routing experiments on the MMLU dataset (Section 4.1). This reasoning query does not route to GPT-3.5-turbo.

Source Sentence (Zh)	没有增设额外的交通工具，地上火车在温布利 (Wembley) 也不停站，地上也没有停车场和停车换乘设施。
Translation (En) (OpenChat-3.5)	There is no additional traffic facility, and the surface train stops at Wembley, and there is no parking lot or interchange facility on the surface.
Translation (En) (GPT-4-turbo)	No additional transportation has been added, ground trains do not stop at Wembley, and there are no parking lots or park-and-ride facilities on the ground.
Source Sentence (Zh)	佩里表示，他会“返回得克萨斯，评估今晚党团会议的结果，明确自己在这次较量中是否有胜算的余地”。但后来他又表示会继续参加竞选，并参加于 1 月 21 日在南卡罗莱纳州举行的初选。
Translation (En) (LLaMA-2)	Perry indicated that he would "return to Kansas and evaluate the results of the party meeting tonight, and make it clear whether he has any advantageous situations in this round of competition." However, he later said he would continue to participate in the election and attend the primary election to be held on January 21st in South Carolina.
Translation (En) (GPT-3.5-turbo)	Perry said he would "return to Texas **to** evaluate the results of tonight's caucus and determine whether he has a chance of winning in this contest." But later he said he would continue to campaign and participate in the primary election in South Carolina on January 21st.
Source Sentence (Zh)	被告的姓名分别是 Baba Kanjar、Bhutha Kanjar、Rampro Kanjar、Gaza Kanjar 和 Vishnu Kanjar。
Translation (En) (OpenChat-3.5)	The names of the defendants are Baba Kanjar, Bhutha Kanjar, Rampro Kanjar, Gaza Kanjar, and Vishnu Kanjar.
translation (En) (GPT-4-turbo)	*The names of the defendants are Baba Kanjar, Bhutha Kanjar, Rampro Kanjar, Gaza Kanjar, and Vishnu Kanjar. (No Improvement)*

Table 12: A few cases in query routing experiments (Section 4.1) on Zh-En translation tasks. All cases route the query from open-source LLMs to API-based ones. While the third case fail to gain improvement.

Filtered	**Instruction**	Split the following string into its components ["Hello", "world", "!"].
	Input	Hello world!
	Output	The components of the string "Hello world!" are ["Hello", "world", "!"].
	Instruction	Extract four highlights from the given article.
	Input	https://www.cnet.com/news/zoom-adds-security-and-encryption-in-latest-update/
	Output	1. Users can now access end-to-end encryption for fully secure video calls. 2. Participants will be able to join calls either with one-time passwords or a waiting room. 3. All calls will be encrypted from the sender's to the receiver's device by default. 4. Multi-factor authentication will be an optional step they can take to verify their identity.
	Instruction	Implement the quick-sort algorithm to sort the following array.
	Input	[2, 4, 3, 7, 5]
	Output	[2, 3, 4, 5, 7]
Refined	**Instruction**	For a given positive integer n, generate all the possible permutations of numbers with length n.
	Input	2
	Output	[1,2], [2,1]
	Instruction	Create an algorithm to check if a number is prime or not.
	Input	7

	Output	def is_prime(number):
		  if number <= 1:
		  return False
		  else:
		  for n in range(2, number):
		  if number % n == 0:
		  return False
		  return True
		
		print(is_prime(7))
		# Output
		True

	Instruction	Given a list of ingredients, describe how to make a delicious pizza.
	Input	Dough, tomatoes, basil, mozzarella, olive oil.
	Output	Begin by preheating your oven to the desired temperature. Roll out the dough on a greased pizza pan. Cover the dough with a thin layer of olive oil. Spread the diced tomatoes and basil leaves on top of the dough. Add the mozzarella cheese over top of the pizza evenly. Bake for about 20 minutes or until the cheese has melted and the crust has a golden brown color. Enjoy your delicious pizza!

Table 13: A few cases of filtered and refined training data from the final epoch in the iterative training data refinement experiment on Phi-2. These samples are used in the stage 2 for pKTO training in Section 4.2. The second sample generate outputs from a website with only the URL and the third did not generate a quick-sort algorithm.

Prompt	Content	Task
MMLU instruction	{Question}\nA. {Choice A}\nB. {Choice B}\nC. {Choice C}\nD. {Choice D}\nPlease think step by step and give the answer.\n	E.D. & Q.R.
MMLU instruction (GPT)	Can you answer the following question? {Question}: A) {Choice A}, B) {Choice B}, C) {Choice C}, D) {Choice D} Explain your answer, putting the answer in the form (X) at the end of your response.	E.D. & Q.R.
CMMLU instruction	以下是关于{Category}的单项选择题，请直接给出正确答案的选项。{Question}\nA. {Choice A}\nB. {Choice B}\nC. {Choice C}\nD. {Choice D}\n请一步步思考并给出答案。	E.D. & Q.R.
CMMLU instruction (GPT)	以下是关于{Category}的单项选择题，请给出正确答案的选项。\n\n{Question}\nA. {Choice A}\nB. {Choice B}\nC. {Choice C}\nD. {Choice D} \n\n请思考后回答，在结尾处的(X)内写上答案的选项。	E.D. & Q.R.
JCommonsenseQA	{Question}\nA. {Choice A}\nB. {Choice B}\nC. {Choice C}\nD. {Choice D}\nE. {Choice E}\n一一考えて答えを出してください。	E.D.
JCommonsenseQA (GPT)	{Question}\n(A) {Choice A}\n(B) {Choice B}\n(C) {Choice C}\n(D) {Choice D}\n(E) {Choice E}\n\n一一考えて、答えを最後に（X）の形でいてください。	E.D.
Zh-En Translation instruction	Translate the following sentence from Chinese to English (only output the translated sentence). \n\n{Zh Sentence}	Q.R.
En-Zh Translation instruction	请将以下句子从英语翻译成中文（直接输出翻译后的句子）。\n\n{En Sentence}	Q.R.
Code Translation instruction	code translation:\nPython:\n{Python declaration + solution}\n Java:\n{Java Function Signature}	Q.R.
P(T)	Based on the question, please judge the given answer's correctness. If the answer is correct, please write 'T', otherwise, please write 'F'.\n\nQuestion: {Question}\n\nAnswer: {Answer}\n\nJudgement (T/F):	E.D.
Meta Ranking	**Question 1:** {Query 1} **Answer 1:** {Response 1} **Question 2:** {Query 2} **Answer 2:** {Response 2} **Evaluation Request:** Please evaluate and compare the correctness of the answers provided for Question 1 and Question 2. Consider the following aspects: 1. **Accuracy:** How accurate are the answers in relation to the questions? Are the facts or information provided correct? 2. **Relevance:** Are the answers relevant to the questions asked? Do they address the main point or topic of the question? 3. **Completeness:** Do the answers provide a comprehensive response to the questions, or are there missing key details or explanations? 4. **Clarity:** Are the answers clear and easy to understand? Do they avoid unnecessary complexity or ambiguity? Based on these criteria, please provide an assessment of which question-and-answer pair is more correct or if they are equally valid, by outputting the number of the pair (1. Q1&A1; 2. Q2&A2; 3. Equally valid).\n	E.D. & Q.R.

Prompt	Content	Task
Meta Ranking	**Instruction 1:** `{Query 1}` **Response 1:** `{Response 1}` **Instruction 2:** `{Query 2}` **Response 2:** `{Response 2}` **Evaluation Request:** Please evaluate and compare the correctness of the response provided for Instruction 1 and Instruction 2. Consider the following aspects: - Relevance to the instruction - Accuracy of information - Clarity of explanation (e.g., readable format) - Completeness of response - Harmlessness of response - Complexity of the instruction Based on these criteria, please provide an assessment of which instruction-and-response pair is better or if they are equally valid, by outputting the number of the pair (1. I1&R1; 2. I2&R2; 3. Equally valid).	I.T.D.R.

Table 14: Instruction and prompt templates used in different datasets and tasks. We use E.D. to denote error detection tasks, Q.R. to denote query routing tasks, and I.T.D.R. to stand for iterative training data refinement.