NOT ALL CRITERIA MATTER: IMPORTANCE-AWARE CRITERIA WEIGHTING FOR RUBRIC-BASED LLM EVALUATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Rubric-based evaluation assesses the responses of large language models (LLMs) by aggregating the verification results of explicit criteria, yet *not all criteria matter equally*. However, existing methods commonly aggregate the verification results with uniform weighting, which assigns minor and overlapping criteria the same weight as essential ones and thus produces misleading evaluation scores. Although some methods prompt an LLM to directly assign weights, direct weighting requires a holistic judgment over all criteria that LLMs struggle to make reliably. To address these limitations, we propose Importance-Aware Criteria Weighting (ICW), which derives the weights of criteria from pairwise importance comparisons made by an LLM. Specifically, ICW transforms the holistic judgment over all criteria into a series of pairwise importance comparisons, checks and revises inconsistent comparisons, and derives the weights from the principal eigenvector of the comparison matrix. Extensive experiments demonstrate that ICW consistently outperforms uniform weighting and competitive weighting methods, and translates into better policy models in downstream reinforcement learning. Code is available at https://anonymous.4open.science/r/ICW-F5F1.

1 INTRODUCTION

Rubric-based evaluation, which assesses response quality against explicit criteria, has emerged as a prevalent paradigm for evaluating open-ended responses of large language models (LLMs) (Hashemi et al., 2024; Li et al., 2025a; Zhang et al., 2026b), with widespread applications in domains such as healthcare (Arora et al., 2025), deep research (Du et al., 2025), and scholarly question answering (Yifei et al., 2026). Specifically, an LLM-based verifier judges whether a response satisfies each criterion, and the verification results are aggregated into an evaluation score.

To improve rubric-based evaluation, existing works mainly follow two lines. One line of work studies rubric generation, where criteria are written by experts (Arora et al., 2025), generated by LLMs for each user prompt (Cook et al., 2024; Li et al., 2026), and further refined to better distinguish between responses (Zhang et al., 2026a; Shen et al., 2026). A further line of work uses the evaluation score as a reward signal in reinforcement learning (RL) for open-ended tasks without verifiable answers (Gunjal et al., 2026; Viswanathan et al., 2025; Shan & Shao, 2026).

Despite the progress in both lines of work, existing methods commonly aggregate the verification results with uniform weighting (Qin et al., 2024a; Cook et al., 2024; Lee et al., 2025), which implicitly assumes that all criteria are equally important. However, *not all criteria matter equally*: some criteria are essential for addressing the user prompt, whereas others only concern minor aspects such as formatting or tone, and several criteria may describe the same property of a response. By assigning minor and overlapping criteria the same weight as essential ones, uniform weighting can produce misleading evaluation scores that favor lower-quality responses. While some methods prompt an LLM to directly assign weights to criteria (Du et al., 2025; Gunjal et al., 2026; Zhang et al., 2026a; Ding, 2026), direct weighting requires joint judgments over the rubric set and numerical weight assignments. Studies of LLM-based text evaluation report anchoring effects in multi-attribute judgments and inconsistent numerical ratings (Stureborg et al., 2024). These findings motivate examining alternatives to uniform weighting and direct numerical weight elicitation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

To this end, we propose Importance-Aware Criteria Weighting (ICW), which derives the weights of criteria from pairwise importance comparisons made by an LLM. This design is motivated by evidence for the effectiveness of pairwise judgments in text evaluation and ranking (Liusie et al., 2024; Liu et al., 2024; Qin et al., 2024b). Specifically, ICW transforms the holistic judgment over all criteria into a series of pairwise importance comparisons, each of which only requires the LLM to judge which of two criteria is more important for the user prompt, and derives the weights from the principal eigenvector of the resulting comparison matrix. Since independent comparisons may contradict one another, ICW further checks the consistency of the comparisons and prompts the LLM to revise inconsistent comparisons. Experimental results confirm that ICW produces more accurate evaluation scores and better policy models in downstream RL.

Contributions. The contributions of this work are summarized as follows. ❶ **We identify the unequal importance of criteria as an under-addressed challenge in rubric-based evaluation.** Since not all criteria matter equally, uniform weighting can produce misleading evaluation scores. ❷ **We propose ICW, which derives the weights of criteria from pairwise importance comparisons made by an LLM.** ICW derives the weights from the principal eigenvector of the comparison matrix, and checks and revises inconsistent comparisons. ❸ **We conduct comprehensive experiments on six pairwise preference benchmarks to evaluate the efficacy of ICW,** showing that ICW consistently outperforms uniform weighting and competitive weighting methods, generalizes across verifiers and LLMs from different model families, and improves downstream RL performance.

2 Preliminaries

2.1 Rubric-Based Evaluation

Rubric-based evaluation assesses response quality against explicit criteria (Hashemi et al., 2024; Arora et al., 2025; Du et al., 2025). Given a user prompt x and an LLM-generated response y, we define a rubric set $\mathcal{R} = \{c_1, \ldots, c_K\}$ comprising K readily verifiable criteria. Each criterion is a textual statement characterizing a property of an ideal response. Suppose $v(\cdot)$ is an LLM-based verifier that outputs $v(x, y, c_k) \in \{0, 1\}$ to indicate whether y satisfies c_k given x. The evaluation score is defined as:

$$s(x, y) = \sum_{k=1}^{K} w_k \, v(x, y, c_k), \tag{1}$$

where $\boldsymbol{w} = (w_1, \ldots, w_K)$ denotes the weights reflecting the relative importance of the criteria, with $w_k \geq 0$ and $\sum_{k=1}^{K} w_k = 1$. By aggregating verification results across criteria, $s(x, y)$ provides an overall measure of response quality for downstream applications. For data curation, $s(x, y)$ can be used to rank candidate responses and select high-quality examples (Li et al., 2026); in RL, it can serve as a reward signal to optimize the policy model (Gunjal et al., 2026; Viswanathan et al., 2025; Zhang et al., 2026a).

2.2 Problem Definition

This work investigates the rubric weighting problem, which aims to estimate the weights $\boldsymbol{w}$ in Eq. (1) that reflect the relative importance of the criteria. Specifically, $\boldsymbol{w}$ is estimated from the user prompt x and the fixed rubric set $\mathcal{R}$ alone, and is shared across candidate responses to aggregate their verification results. **Thus, the core problem is to estimate $\boldsymbol{w}$ from x and $\mathcal{R}$ such that $s(x, y)$ ranks candidate responses consistently with human preference.**

3 Methodology

3.1 Motivation

Rubric weighting is central to rubric-based evaluation. Given the verification results, the weights $\boldsymbol{w}$ in Eq. (1) determine how candidate responses are ranked, and thus which responses are selected in data curation (Li et al., 2026) and rewarded in RL (Gunjal et al., 2026). Therefore, accurate rubric weighting is essential for reliable rubric-based evaluation and its downstream applications.

A simple solution to the rubric weighting problem is uniform weighting, which assigns $w_k = 1/K$ to every criterion. However, *not all criteria matter equally.* Specifically, some criteria are essential for addressing the user prompt, whereas others only concern minor aspects of a response, such as formatting or tone, and several criteria may describe the same property of a response. By assigning minor and overlapping criteria the same weight as essential ones, uniform weighting allows several minor criteria to jointly outweigh an essential criterion and overlapping criteria to give repeated credit to the same property. Consequently, a lower-quality response may receive a higher evaluation score than a higher-quality response.

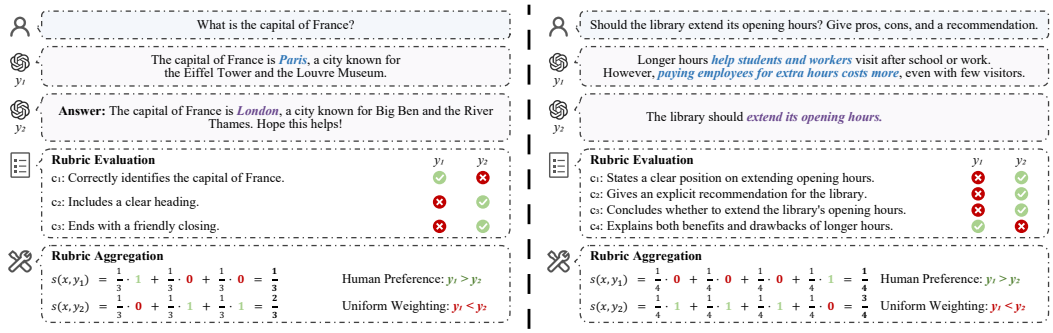

Figure 1: Case studies in which uniform weighting assigns higher evaluation scores to lower-quality responses. (a) Minor stylistic criteria outweigh the essential criterion of factual correctness. (b) Overlapping criteria give repeated credit to the same recommendation.

Case studies. To further substantiate the above limitation, we conduct case studies as follows. In Figure 1(a), several minor criteria outweigh an essential criterion: although y_1 correctly answers that the capital of France is Paris, the incorrect y_2 receives a higher evaluation score because y_2 includes a heading and a friendly closing. In Figure 1(b), overlapping criteria give repeated credit to the same property: although y_1 explains both benefits and drawbacks of longer hours, the less informative y_2 receives a higher evaluation score because the recommendation of y_2 satisfies three overlapping criteria. **Therefore, since not all criteria matter equally, uniform weighting can produce misleading evaluation scores that favor lower-quality responses.**

3.2 Pairwise Importance Comparison

To account for the unequal importance of criteria, a natural approach is to prompt an LLM to directly assign numerical weights or importance levels to criteria (Du et al., 2025; Viswanathan et al., 2025; Gunjal et al., 2026; Zhang et al., 2026a; Ding, 2026; Li et al., 2026). This approach requires the LLM to express its judgments over the rubric set as weights or importance levels in a single response. Studies of LLM-based text evaluation report anchoring effects in multi-attribute judgments and inconsistent numerical ratings (Stureborg et al., 2024). Prior work also reports improvements from pairwise comparisons over absolute scoring in text evaluation (Liusie et al., 2024; Liu et al., 2024). Motivated by these findings, we transform the holistic judgment over all criteria into a series of pairwise importance comparisons, each of which only requires judging which of two criteria is more important.

To this end, given the user prompt x and the rubric set $\mathcal{R}$, we prompt an LLM, referred to as the weighting model, to judge which of two criteria $c_i, c_j \in \mathcal{R}$ is more important for evaluating responses to x, together with a brief reason. The weighting model may also judge that neither criterion is more important when the two criteria overlap or are equally important. Formally, we denote the judgment when c_i is displayed before c_j by $o_{ij} = \mathrm{Compare}(x, \mathcal{R}, c_i, c_j) \in \{i, j, 0\}$, where $o_{ij} = i$ or $o_{ij} = j$ indicates the more important criterion and $o_{ij} = 0$ indicates that neither is more important.

However, LLM judgments are sensitive to the order in which two candidates are displayed (Zheng et al., 2023). To mitigate this position bias, we query each pair in both display orders and regard a criterion as more important only when the two judgments agree. We use a positive reciprocal

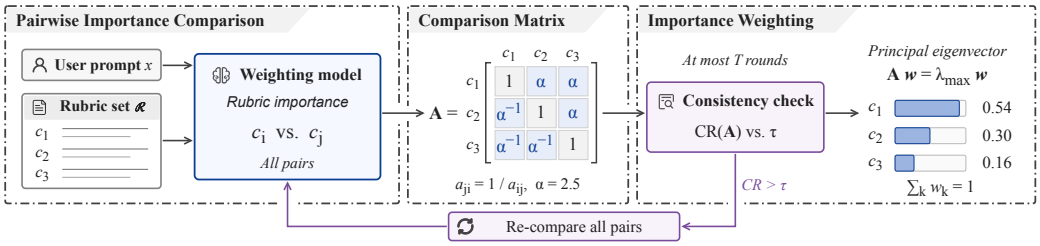

Figure 2: Overview of ICW. The weighting model compares the importance of every pair of criteria given the user prompt and the rubric set, and the judgments are mapped to a comparison matrix with the fixed ratio α. If CR exceeds τ, all pairs are re-compared for at most T rounds, and a revised matrix replaces the current matrix only if its CR is strictly lower. The weights are obtained from the principal eigenvector of the current matrix.

comparison matrix from AHP (Saaty, 1977). To populate it from ordinal judgments, we encode each accepted comparison with a fixed ratio $\alpha > 1$, so the weighting model only needs to identify the more important criterion without estimating a numerical intensity. Combining the two designs, we define the comparison of c_i against c_j as:

$$a_{ij} = \begin{cases} \alpha, & \text{if } o_{ij} = o_{ji} = i, \\ 1/\alpha, & \text{if } o_{ij} = o_{ji} = j, \\ 1, & \text{otherwise}, \end{cases} \tag{2}$$

with $a_{ii} = 1$. In particular, $a_{ij} = 1$ when neither criterion is judged more important or when the two display orders disagree, so that an order-dependent judgment favors neither criterion. The resulting comparison matrix $\mathbf{A} = (a_{ij}) \in \mathbb{R}_{>0}^{K\times K}$ is positive and reciprocal, i.e., $a_{ji} = 1/a_{ij}$, and summarizes the relative importance between every pair of criteria for x.

3.3 Importance Weighting via Principal Eigenvector

In this section, we derive the weights $\boldsymbol{w}$ from the comparison matrix $\mathbf{A}$. The key idea is that a_{ij} specifies how the weight of c_i should relate to that of c_j; for example, $a_{ij} = \alpha$ means that c_i should receive α times the weight of c_j. Therefore, each comparison should ideally equal the ratio between the weights of the two criteria, i.e., $a_{ij} = w_i/w_j$. Since $(w_i/w_j)(w_j/w_\ell) = w_i/w_\ell$, the ratio $a_{ij} = w_i/w_j$ can hold for all pairs only if the direct comparison of any two criteria agrees with their indirect comparison through any other criterion. Formally, we call $\mathbf{A}$ consistent if $a_{i\ell} = a_{ij}a_{j\ell}$ for all i, j, ℓ. Proposition 1 shows that consistency is also sufficient for such weights to exist, and that the weights are then given by the principal eigenvector of $\mathbf{A}$.

Proposition 1 (Saaty, 1977)**.** *A positive reciprocal matrix $\mathbf{A} \in \mathbb{R}_{>0}^{K\times K}$ is consistent if and only if there exist weights $\boldsymbol{w}$ with $w_k > 0$ and $\sum_{k=1}^{K} w_k = 1$ such that $a_{ij} = w_i/w_j$ for all i, j. In this case, $\boldsymbol{w}$ is unique and is the principal eigenvector of $\mathbf{A}$ with eigenvalue $\lambda_{\max} = K$.*

The proof is provided in Appendix B. Therefore, it is natural to obtain the weights from the principal eigenvector of $\mathbf{A}$:

$$\mathbf{A}\boldsymbol{w} = \lambda_{\max}\boldsymbol{w}, \qquad \sum_{k=1}^{K} w_k = 1, \qquad w_k > 0, \tag{3}$$

where $\lambda_{\max}$ is the largest eigenvalue of $\mathbf{A}$. By the Perron–Frobenius theorem (Horn & Johnson, 2012), $\boldsymbol{w}$ exists and is unique for any positive matrix, so the weights are well defined even when $\mathbf{A}$ is inconsistent. Moreover, by Proposition 1, when the comparisons are consistent, a criterion judged more important than another receives α times the weight of the other criterion, and $\alpha = 1$ recovers uniform weighting. In this way, the weights aggregate the pairwise importance comparisons of all criteria, so that criteria judged more important receive larger weights and the evaluation score in Eq. (1) no longer treats all criteria equally.

3.4 Consistency Check and Revision

Although the principal eigenvector yields weights from any comparison matrix, the weights reproduce all comparisons, i.e., $a_{ij} = w_i/w_j$ for all i, j, only when $\mathbf{A}$ is consistent (Proposition 1). Since each pair is compared separately, the comparisons may be inconsistent with each other. For example, if c_i is judged more important than c_j, c_j more important than c_ℓ, and c_ℓ more important than c_i, no weights can reproduce the three comparisons, and for $K = 3$ the principal eigenvector assigns equal weights to the three criteria although every comparison favors one of them. Inconsistent comparisons thus indicate unreliable judgments, under which the weights may misrepresent the relative importance of the criteria. In this section, we introduce consistency check and revision to detect and revise inconsistent comparisons. The consistency check builds on Proposition 2, which characterizes consistency through the largest eigenvalue $\lambda_{\max}$.

Proposition 2 (Saaty, 1977)**.** *For any positive reciprocal matrix $\mathbf{A} \in \mathbb{R}^{K\times K}_{>0}$, $\lambda_{\max} \geq K$, with equality if and only if $\mathbf{A}$ is consistent.*

The proof is provided in Appendix B. Therefore, the gap between $\lambda_{\max}$ and K quantifies the inconsistency of $\mathbf{A}$. Following Saaty (1977), we normalize the gap into the consistency index (CI), which can be rewritten as (see Appendix B):

$$\mathrm{CI}(\mathbf{A}) = \frac{\lambda_{\max} - K}{K-1} = \frac{1}{K(K-1)} \sum_{i<j} \left(\epsilon_{ij} + \frac{1}{\epsilon_{ij}} - 2 \right), \qquad \epsilon_{ij} = \frac{a_{ij} w_j}{w_i}, \tag{4}$$

where $\boldsymbol{w}$ is the principal eigenvector in Eq. (3). Since $\epsilon_{ij} + 1/\epsilon_{ij} - 2 \geq 0$ with equality if and only if $a_{ij} = w_i/w_j$, CI aggregates the deviations between the comparisons and the ratios of the obtained weights over all pairs, i.e., CI measures how well the weights reproduce the comparisons. Since the CI of random comparisons grows with K, we further normalize CI into the consistency ratio (CR), so that one threshold applies to rubric sets of different sizes:

$$\mathrm{CR}(\mathbf{A}) = \frac{\mathrm{CI}(\mathbf{A})}{\mathrm{RI}_K}, \tag{5}$$

where RI_K is the average CI of randomly generated positive reciprocal matrices of size K (Saaty, 1980) (see Appendix D.2 for $K > 15$).[1] By Proposition 2, $\mathrm{CR}(\mathbf{A}) = 0$ if and only if $\mathbf{A}$ is consistent. We accept $\mathbf{A}$ if $\mathrm{CR}(\mathbf{A}) \leq \tau$, where τ is the consistency threshold.

Otherwise, we revise the comparisons for at most T rounds, starting from $\mathbf{A}$ as the current matrix. In each round, the weighting model re-compares every pair in both display orders given the judgments and reasons of the previous round, denoted by $\mathcal{H}$, i.e., $o'_{ij} = \mathrm{Compare}(x, \mathcal{R}, c_i, c_j; \mathcal{H})$, so that each pair is re-examined in light of the judgments of the other pairs. The revised judgments form a revised matrix $\mathbf{A}'$ via Eq. (2), which replaces the current matrix $\mathbf{A}$ only if $\mathrm{CR}(\mathbf{A}') < \mathrm{CR}(\mathbf{A})$. The revision stops once $\mathrm{CR}(\mathbf{A}) \leq \tau$ or after T rounds, and the weights are obtained from the current matrix $\mathbf{A}$ via Eq. (3). In this way, the revision never increases the inconsistency of the comparisons, and the weights are derived from comparisons that more reliably reflect the relative importance of the criteria.

3.5 The Workflow of ICW

In this section, we present the workflow of ICW, as illustrated in Figure 2 and Algorithm 1. First, the weighting model compares every pair of criteria in both display orders (steps 1–3), and the judgments are mapped to the comparison matrix $\mathbf{A}$ via Eq. (2) (step 4). Then, while CR exceeds τ, all pairs are re-compared given the judgments and reasons of the previous round for at most T rounds, and each revised matrix replaces the current matrix only if its CR is strictly lower (steps 5–9). Finally, the weights are obtained from the principal eigenvector of the current matrix via Eq. (3) (step 10). Each round involves $K(K-1)$ queries to the weighting model, which are independent of each other and can be issued in parallel. Since $\boldsymbol{w}$ depends only on x and $\mathcal{R}$, it is computed once for each user prompt and shared across all candidate responses, so that the cost of ICW does not grow with the number of candidate responses.

[1] For $K \leq 2$, every positive reciprocal matrix is consistent, and we set $\mathrm{CR}(\mathbf{A}) = 0$.

Algorithm 1 The workflow of ICW

Input: user prompt x; rubric set $\mathcal{R} = \{c_1, \ldots, c_K\}$.
Parameter: fixed ratio α; consistency threshold τ; maximum rounds T.
Output: weights $\boldsymbol{w}$.

for each pair (c_i, c_j) with $i < j$ **do**
  $o_{ij} \leftarrow \mathrm{Compare}(x, \mathcal{R}, c_i, c_j)$
  $o_{ji} \leftarrow \mathrm{Compare}(x, \mathcal{R}, c_j, c_i)$
Construct $\mathbf{A}$ via Eq. (2); $t \leftarrow 0$
while $\mathrm{CR}(\mathbf{A}) > \tau$ **and** $t < T$ **do**
  $t \leftarrow t + 1$
  Re-compare all pairs given $\mathcal{H}$ (steps 1–3)
  Construct $\mathbf{A}'$ via Eq. (2)
  if $\mathrm{CR}(\mathbf{A}') < \mathrm{CR}(\mathbf{A})$ **then** $\mathbf{A} \leftarrow \mathbf{A}'$
Obtain $\boldsymbol{w}$ from $\mathbf{A}$ via Eq. (3)
return $\boldsymbol{w}$

4 EXPERIMENTS

In this section, we conduct a systematic empirical evaluation to demonstrate the efficacy of ICW, covering five research questions (RQs):

1. **RQ1**: *Does ICW perform well?* In Section 4.2, we compare ICW against uniform weighting and competitive weighting methods on six benchmarks.
2. **RQ2**: *Why does it work?* In Section 4.3, we perform an ablation study on the contribution of pairwise importance comparison and consistency check and revision.
3. **RQ3**: *Does it generalize across verifiers and weighting models?* In Section 4.4, we evaluate its compatibility with different verifiers and weighting models.
4. **RQ4**: *Is it sensitive to hyperparameters?* In Section 4.5, we analyze its sensitivity to α and τ.
5. **RQ5**: *Does it improve downstream RL?* In Section 4.6, we validate whether the improved evaluation scores translate into better policy models.

4.1 EXPERIMENTAL SETUP

Datasets and Rubrics. We evaluate on six pairwise preference benchmarks, where each sample consists of a user prompt, two candidate responses, and a preference label: RubricBench (Zhang et al., 2026b), ResearchQA (Yifei et al., 2026), JudgeBench (Tan et al., 2025), the Chat subset of RM-Bench (Liu et al., 2025) (RM-Chat), the Chat-Hard subset of RewardBench (Lambert et al., 2025) (Chat-Hard), and the Precise-IF subset of RewardBench 2 (Malik et al., 2026) (Precise-IF). For RubricBench and ResearchQA, we use the rubric sets provided by the datasets. For the other four benchmarks, we use the rubric sets generated by the rubric generator of Rubric-ARM (Xu et al., 2026). The weights provided by the rubric sources are removed, so that all methods weight the same criteria for each user prompt. More details are provided in Appendix D.1.

Baselines. We compare ICW against diverse rubric weighting methods, grouped into: (1) **Direct Weighting Methods**, which prompt an LLM to assign weights or importance levels to all criteria at once, including Direct, AdaRubric (Ding, 2026), RaR (Gunjal et al., 2026), RACE-SC (Du et al., 2025), and ChasingTail (Zhang et al., 2026a); and (2) **Indirect Weighting Methods**, which derive the weights from pairwise comparisons or verification results, including MAPLE (Jinji et al., 2026) and RRD-WU (Shen et al., 2026), respectively. Except for Direct, these methods are originally proposed within larger pipelines, such as rubric generation and policy training, and we adopt only their weighting components. We further report Uniform, which assigns equal weights to all criteria, as a reference.

Evaluation Metric. For each sample, a method predicts the candidate response with the higher evaluation score in Eq. (1) as the preferred one, where a tie earns half credit. We report the pairwise preference accuracy (Acc), together with the relative improvement (RI) over Uniform.

Implementation Details. For fair comparison, all methods share the same rubric sets and verification results, and differ only in the weights. Unless otherwise specified, we use GPT-4.1 as the

Table 1: Pairwise preference accuracy of ICW versus baselines.

Method	RubricBench		ResearchQA		JudgeBench		RM-Chat		Chat-Hard		Precise-IF	
	Acc	RI	Acc	RI	Acc	RI	Acc	RI	Acc	RI	Acc	RI
Uniform	0.771	–	0.583	–	0.762	–	0.740	–	0.805	–	0.720	–
Direct	0.778	0.96% ↑	0.589	1.03% ↑	0.770	1.06% ↑	0.731	1.22% ↓	0.802	0.41% ↓	0.723	0.43% ↑
AdaRubric	0.774	0.45% ↑	0.592	1.53% ↑	0.765	0.31% ↑	0.749	1.23% ↑	0.812	0.96% ↑	0.732	1.74% ↑
RaR	0.782	1.41% ↑	0.604	3.57% ↑	0.763	0.10% ↑	0.760	2.62% ↑	0.816	1.37% ↑	0.746	3.61% ↑
MAPLE	0.782	1.41% ↑	0.601	3.05% ↑	0.760	0.21% ↓	0.745	0.70% ↑	0.812	0.96% ↑	0.738	2.60% ↑
RRD-WU	0.780	1.25% ↑	0.610	4.58% ↑	0.772	1.27% ↑	0.760	2.62% ↑	0.815	1.23% ↑	0.744	3.33% ↑
ChasingTail	0.782	1.47% ↑	0.604	3.57% ↑	0.770	1.06% ↑	0.760	2.62% ↑	0.814	1.09% ↑	0.743	3.18% ↑
RACE-SC	0.780	1.19% ↑	0.601	3.05% ↑	0.768	0.84% ↑	0.757	2.27% ↑	0.810	0.68% ↑	0.756	5.07% ↑
ICW	**0.795**	**3.11% ↑**	**0.612**	**5.08% ↑**	**0.783**	**2.76% ↑**	**0.769**	**3.84% ↑**	**0.818**	**1.64% ↑**	**0.762**	**5.79% ↑**

Note: "RI" denotes the relative improvement over the Uniform baseline.

Table 2: Ablation study results of the two ICW components.

Method	Pair.	Rev.	RubricBench		ResearchQA		JudgeBench		RM-Chat		Chat-Hard		Precise-IF	
			Acc	RI	Acc	RI	Acc	RI	Acc	RI	Acc	RI	Acc	RI
Uniform	✗	✗	0.771	–	0.583	–	0.762	–	0.740	–	0.805	–	0.720	–
ICW†	✗	✓	0.782	1.47% ↑	0.592	1.53% ↑	0.758	0.52% ↓	0.747	0.88% ↑	0.803	0.27% ↓	0.755	4.92% ↑
ICW‡	✓	✗	0.790	2.49% ↑	0.607	4.07% ↑	**0.783**	**2.76% ↑**	0.758	2.44% ↑	0.804	0.14% ↓	0.752	4.49% ↑
ICW	✓	✓	**0.795**	**3.11% ↑**	**0.612**	**5.08% ↑**	**0.783**	**2.76% ↑**	**0.769**	**3.84% ↑**	**0.818**	**1.64% ↑**	**0.762**	**5.79% ↑**

Note: "Pair." denotes pairwise importance comparison, and "Rev." denotes consistency check and revision.

weighting model for ICW and the LLM-based baselines, and Gemini-3.1-Flash-Lite as the verifier. Both the weighting model and the verifier are queried with temperature 0. For ICW, we set the maximum number of revision rounds to $T = 1$, and select the fixed ratio α and the consistency threshold τ for each benchmark, as listed in Appendix D.2.

4.2 Overall Performance (RQ1)

In this section, we compare ICW against uniform weighting and competitive weighting methods on six benchmarks. The results are presented in Table 1 with key observations as follows. ❶ **Uniform weighting leads to inferior evaluation accuracy.** Uniform underperforms nearly all weighting methods across benchmarks. This indicates that treating all criteria equally produces misleading evaluation scores, since not all criteria matter equally. ❷ **Direct weighting methods offer limited and inconsistent improvements.** Direct only marginally improves over Uniform and even underperforms Uniform on RM-Chat and Chat-Hard, while stabilizing the direct weights by averaging repeated generations (RACE-SC) or assigning coarse importance levels (RaR and ChasingTail) brings larger improvements. This indicates that the holistic judgment over all criteria is unreliable, which limits the accuracy of direct weighting. ❸ **Indirect weighting methods partially address the limitation of uniform weighting.** RRD-WU achieves the strongest baseline performance by downweighting correlated criteria. However, it disregards the relative importance of the criteria, leading to suboptimal performance. ❹ **ICW achieves state-of-the-art performance across all benchmarks.** ICW consistently outperforms all baselines, improving the accuracy over Uniform by up to 5.79%. This success is attributed to pairwise importance comparison and consistency check and revision, which together yield weights that reflect the relative importance of the criteria.

4.3 Ablation Study (RQ2)

In this section, we dissect the individual contributions of the two designs of ICW: pairwise importance comparison, and consistency check and revision. The results are presented in Table 2 with key observations as follows. ❶ **Pairwise importance comparison yields the dominant performance gain.** Specifically, in ICW†, we compare all criterion pairs within a single query for each display order, while retaining consistency check and revision. We observe a consistent performance drop across all benchmarks (e.g., the accuracy decreases from 0.783 to 0.758 on JudgeBench). This confirms that comparing all pairs at once reintroduces the holistic judgment over all criteria, which is less reliable than judging which of two criteria is more important. ❷ **Consistency check and revi-**

Table 3: Results across verifiers.

Method	RM-Chat		Chat-Hard		Precise-IF	
	Acc	RI	Acc	RI	Acc	RI
Verifier: Gemini-3.5-Flash						
Uniform	0.739	–	0.777	–	0.728	–
ICW	**0.742**	**0.35% ↑**	**0.819**	**5.36% ↑**	**0.767**	**5.30% ↑**
Verifier: GPT-4.1-mini						
Uniform	0.668	–	0.794	–	0.647	–
ICW	**0.673**	**0.76% ↑**	**0.810**	**2.07% ↑**	**0.655**	**1.28% ↑**
Verifier: DeepSeek-V4-Flash						
Uniform	0.703	–	0.788	–	0.719	–
ICW	**0.715**	**1.66% ↑**	**0.812**	**3.06% ↑**	**0.748**	**4.05% ↑**
Verifier: GLM-5.1						
Uniform	0.712	–	0.774	–	0.734	–
ICW	**0.735**	**3.26% ↑**	**0.820**	**5.96% ↑**	**0.769**	**4.68% ↑**
Verifier: Gemini-3.1-Flash-Lite						
Uniform	0.740	–	0.805	–	0.720	–
ICW	**0.769**	**3.84% ↑**	**0.818**	**1.64% ↑**	**0.762**	**5.79% ↑**

Note: "RI" denotes relative improvement over the Uniform baseline.

Table 4: Results across weighting models.

Method	RM-Chat		Chat-Hard		Precise-IF	
	Acc	RI	Acc	RI	Acc	RI
Weighting Model: Gemini-3.5-Flash						
Uniform	0.740	–	0.805	–	0.720	–
ICW	**0.758**	**2.44% ↑**	**0.823**	**2.32% ↑**	**0.770**	**6.95% ↑**
Weighting Model: GPT-4.1-mini						
Uniform	0.740	–	0.805	–	0.720	–
ICW	**0.758**	**2.44% ↑**	**0.820**	**1.91% ↑**	**0.750**	**4.20% ↑**
Weighting Model: DeepSeek-V4-Flash						
Uniform	0.740	–	0.805	–	0.720	–
ICW	**0.760**	**2.62% ↑**	**0.818**	**1.64% ↑**	**0.767**	**6.52% ↑**
Weighting Model: GLM-5.1						
Uniform	0.740	–	0.805	–	0.720	–
ICW	**0.756**	**2.09% ↑**	**0.822**	**2.19% ↑**	**0.768**	**6.65% ↑**
Weighting Model: GPT-4.1						
Uniform	0.740	–	0.805	–	0.720	–
ICW	**0.769**	**3.84% ↑**	**0.818**	**1.64% ↑**	**0.762**	**5.79% ↑**

Note: "RI" denotes relative improvement over the Uniform baseline.

sion yields essential complementary gains. Specifically, in ICW‡, we remove consistency check and revision, and obtain the weights from the initial comparisons. We also observe a performance drop on nearly all benchmarks (e.g., the accuracy decreases from 0.818 to 0.804 on Chat-Hard). This is consistent with Section 3.4: revising inconsistent comparisons allows the weights to better reflect the relative importance of the criteria. ❸ **The benefits from both designs can be synergistically integrated.** This is evidenced by the complete ICW achieving the best accuracy on all benchmarks.

4.4 Generalization Studies (RQ3)

In this section, we evaluate the generalizability of ICW across verifiers and weighting models from different model families, including Gemini, GPT, DeepSeek, and GLM. The results are presented in Tables 3 and 4, and the results on the other three benchmarks are provided in Appendix C.1, with key observations as follows. ❶ **ICW generalizes across verifiers.** With GPT-4.1 as the weighting model, ICW generally outperforms Uniform across the five verifiers. This indicates that the weights of ICW complement the verification results rather than depend on a particular verifier. ❷ **ICW generalizes across weighting models.** With Gemini-3.1-Flash-Lite as the verifier, ICW consistently outperforms Uniform with all five weighting models, including the smaller GPT-4.1-mini. This shows that pairwise importance comparison does not rely on a particular weighting model. Collectively, these findings establish ICW as a model-agnostic rubric weighting method.

4.5 Hyperparameter Sensitivity Study (RQ4)

In this section, we examine the impact of the fixed ratio α and the consistency threshold τ on ICW. The results are presented in Figure 3, and the results with other weighting models are provided in Appendix C.2, with key observations as follows. ❶ **ICW is robust to the fixed ratio.** ICW substantially improves over uniform weighting ($\alpha = 1$) once $\alpha \geq 2$, and the accuracy remains stable as α further increases. Notably, with any tested $\alpha \geq 2$, ICW outperforms the strongest baseline RRD-WU in the average accuracy over the four benchmarks in Figure 3. This indicates that the advantage of ICW stems from pairwise importance comparison rather than a delicate choice of α. ❷ **Smaller consistency thresholds improve accuracy by triggering more revisions.** As τ decreases, more comparison matrices are revised and the accuracy generally improves, whereas ICW approaches its variant without revision for large τ. This is consistent with the ablation results in Section 4.3.

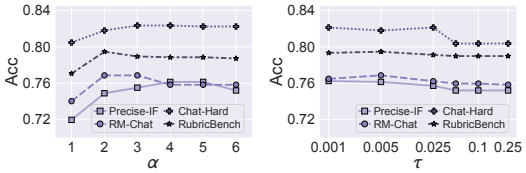

Figure 3: Sensitivity to the fixed ratio α and the consistency threshold τ with GPT-4.1 as the weighting model.

Table 5: Downstream RL results.

Method	BiGGen Bench		AlpacaEval 2.0				Arena-Hard			
	Score	RI	WR	RI	LC	RI	WR	RI	SC	RI
Policy Model: Qwen2.5-3B-Instruct										
Base Model	59.22	–	17.25	–	16.71	–	25.93	–	27.63	–
Uniform	61.45	3.77% ↑	21.59	25.16% ↑	19.07	14.12% ↑	27.58	6.36% ↑	28.58	3.44% ↑
RACE-SC	**64.95**	**9.68%** ↑	24.32	40.99% ↑	22.74	36.09% ↑	28.67	10.57% ↑	29.61	7.17% ↑
RRD-WU	64.23	8.46% ↑	24.37	41.28% ↑	20.42	22.20% ↑	27.78	7.13% ↑	29.80	7.85% ↑
ICW	64.78	9.39% ↑	**26.23**	**52.06%** ↑	**23.54**	**40.87%** ↑	**31.65**	**22.06%** ↑	**31.51**	**14.04%** ↑
Policy Model: Qwen2.5-7B-Instruct										
Base Model	68.98	–	27.79	–	30.08	–	51.90	–	53.41	–
Uniform	71.83	4.13% ↑	40.27	44.91% ↑	33.83	12.47% ↑	56.20	8.29% ↑	56.88	6.50% ↑
RACE-SC	72.12	4.55% ↑	40.42	45.45% ↑	36.19	20.31% ↑	57.08	9.98% ↑	57.68	7.99% ↑
RRD-WU	72.48	5.07% ↑	41.54	49.48% ↑	36.48	21.28% ↑	55.49	6.92% ↑	55.99	4.83% ↑
ICW	**73.84**	**7.05%** ↑	**42.83**	**54.12%** ↑	**37.66**	**25.20%** ↑	**59.78**	**15.18%** ↑	**58.05**	**8.69%** ↑
Policy Model: Qwen3-4B										
Base Model	76.79	–	51.87	–	42.14	–	73.52	–	72.27	–
Uniform	77.73	1.22% ↑	54.37	4.82% ↑	43.70	3.70% ↑	74.46	1.28% ↑	74.21	2.68% ↑
RACE-SC	79.28	3.24% ↑	56.43	8.79% ↑	45.14	7.12% ↑	74.28	1.03% ↑	73.32	1.45% ↑
RRD-WU	79.17	3.10% ↑	56.87	9.64% ↑	45.81	8.71% ↑	75.64	2.88% ↑	74.44	3.00% ↑
ICW	**79.80**	**3.92%** ↑	**56.98**	**9.85%** ↑	**46.39**	**10.09%** ↑	**76.46**	**4.00%** ↑	**75.21**	**4.07%** ↑

Note: "WR", "LC", and "SC" denote the win rate, the length-controlled win rate, and the style-controlled win rate, respectively, and "RI" denotes relative improvement over Base Model.

4.6 Downstream RL Performance (RQ5)

In this section, we investigate whether the improved evaluation scores of ICW, which result from accounting for the unequal importance of criteria, translate into better policy models in downstream RL. We select representative baselines, use the evaluation scores of each method as rewards on user prompts from WildChat (Zhao et al., 2024), and fine-tune three policy models from the Qwen2.5 (Qwen et al., 2024) and Qwen3 (Yang et al., 2025) families via GRPO (Shao et al., 2024). We further report the policy models before RL, referred to as Base Model, as a reference. We evaluate the policy models on BiGGen Bench (Kim et al., 2025), AlpacaEval 2.0 (Dubois et al., 2024), and Arena-Hard (Li et al., 2025b), and more details are provided in Appendix D.2. The results are presented in Table 5 with key observations as follows. ❶ **Uniform weighting also limits its downstream RL performance.** RACE-SC and RRD-WU, which achieve higher accuracy than Uniform in Table 1, generally outperform Uniform. This indicates that treating all criteria equally produces misleading rewards, which in turn limit policy optimization. ❷ **ICW achieves the best overall downstream RL performance.** ICW achieves the best results on nearly all benchmarks across policy models; for example, it improves the AlpacaEval LC win rate of Qwen2.5-3B-Instruct by 40.87% over Base Model, compared with 14.12% for Uniform. These results demonstrate that weights reflecting the relative importance of the criteria provide more effective rewards.

5 Conclusion

In this paper, we investigated the rubric weighting problem in rubric-based evaluation, where not all criteria matter equally but existing methods commonly aggregate the verification results with uniform weighting. To address this limitation, we proposed ICW, which derives the weights of criteria from pairwise importance comparisons made by an LLM, and checks and revises inconsistent comparisons. Extensive experiments confirm that ICW consistently outperforms uniform weighting and competitive weighting methods, improving the pairwise preference accuracy over uniform weighting by up to 5.79%, and translates into better policy models in downstream RL—including a 40.87% improvement in the AlpacaEval LC win rate of Qwen2.5-3B-Instruct over Base Model, compared with 14.12% for uniform weighting.

Limitations & Future Work. Two limitations of this work warrant future investigation: (1) ICW compares every pair of criteria, so the number of queries grows quadratically with the number of criteria (Appendix C.3), motivating future work on comparing only informative pairs. (2) In specialized domains such as healthcare, the weighting model may lack the knowledge required to judge the importance of criteria. Incorporating expert-specified comparisons remains a valuable direction.

AI USE STATEMENT

We used generative AI tools only for auxiliary tasks that fall under the recommended-disclosure category of the ICLR 2027 AI Policy for Authors. These tasks were: (i) copy-editing author-written text for grammar and consistent style, (ii) suggesting candidate references for the related-work section, and (iii) checking author-written code snippets for obvious bugs. We did not use generative AI for any activity that requires mandatory disclosure under that policy. The human authors carried out the research ideation, the formulation of the method, the experimental design, the interpretation of results, and the theoretical proofs. The authors manually reviewed and validated every AI-assisted output before including it. This included independently checking each suggested reference and code change. The authors take full responsibility for the final content of this paper, including its text, its claims, and any released artifacts produced with the help of generative AI.

REFERENCES

Rahul K. Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, Johannes Heidecke, and Karan Singhal. HealthBench: Evaluating large language models towards improved human health. *arXiv preprint arXiv:2505.08775*, 2025. URL https://arxiv.org/abs/2505.08775.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. doi: 10.2307/2334029. URL https://doi.org/10.2307/2334029.

Yifan Chen, Haitao Li, Qingyao Ai, Fengbin Zhu, Tat-Seng Chua, Min Zhang, and Yiqun Liu. GenRubric: Self-evolving rubric generation for scalable LLM evaluation. *arXiv preprint arXiv:2608.29856*, 2026. URL https://arxiv.org/abs/2608.29856.

Jonathan Cook, Tim Rocktäschel, Jakob Foerster, Dennis Aumiller, and Alex Wang. TICKing all the boxes: Generated checklists improve LLM evaluation and generation. *arXiv preprint arXiv:2410.03608*, 2024. URL https://arxiv.org/abs/2410.03608.

Liang Ding. AdaRubric: Task-adaptive rubrics for LLM agent evaluation. *arXiv preprint arXiv:2603.21362*, 2026. URL https://arxiv.org/abs/2603.21362v1.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. DeepResearch Bench: A comprehensive benchmark for deep research agents. *arXiv preprint arXiv:2506.11763*, 2025. URL https://arxiv.org/abs/2506.11763.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B. Hashimoto. Length-controlled AlpacaEval: A simple way to debias automatic evaluators. In *Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2404.04475.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2507.17746.

Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. LLM-rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13806–13834. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.745. URL https://aclanthology.org/2024.acl-long.745/.

Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 2 edition, 2012. doi: 10.1017/CBO9781139020411. URL https://doi.org/10.1017/CBO9781139020411.

Michinori Jinji, Kyohei Atarashi, Koh Takeuchi, and Hisashi Kashima. MAPLE: Multi-aspect panels of LLM evaluators for open-ended questions. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 27071–27088. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1351. URL https://aclanthology.org/2026.findings-acl.1351/.

Seungone Kim, Juyoung Suk, Ji Yong Cho, Shayne Longpre, Chaeeun Kim, et al. The BiGGen bench: A principled benchmark for fine-grained evaluation of language models with language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 2025. URL https://arxiv.org/abs/2406.05761.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. RewardBench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1755–1797, 2025. URL https://aclanthology.org/2025.findings-naacl.96/.

Yukyung Lee, JoongHoon Kim, Jaehee Kim, Hyowon Cho, Jaewook Kang, Pilsung Kang, and Najoung Kim. CheckEval: A reliable LLM-as-a-judge framework for evaluating text generation using checklists. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 15771–15798. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.796. URL https://aclanthology.org/2025.emnlp-main.796/.

Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, et al. From generation to judgment: Opportunities and challenges of LLM-as-a-judge. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 2757–2791, 2025a.

Sunzhu Li, Jiale Zhao, Miteto Wei, Huimin Ren, Yang Zhou, Jingwen Yang, Shunyu Liu, Kaike Zhang, and Wei Chen. RubricHub: A comprehensive and highly discriminative rubric dataset via automated coarse-to-fine generation. *arXiv preprint arXiv:2601.08430*, 2026. URL https://arxiv.org/abs/2601.08430v2.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-Hard and BenchBuilder pipeline. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, 2025b. URL https://proceedings.mlr.press/v267/li25h.html.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. RM-Bench: Benchmarking reward models of language models with subtlety and style. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.16184.

Yinhong Liu, Han Zhou, Zhijiang Guo, Ehsan Shareghi, Ivan Vulić, Anna Korhonen, and Nigel Collier. Aligning with human judgement: The role of pairwise preference in large language model evaluators. In *Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2403.16950.

Adian Liusie, Potsawee Manakul, and Mark Gales. LLM comparative assessment: Zero-shot NLG evaluation through pairwise comparisons using large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 139–151, 2024. URL https://aclanthology.org/2024.eacl-long.8/.

Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A. Smith, Hannaneh Hajishirzi, and Nathan Lambert. RewardBench 2: Advancing reward model evaluation. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2506.01937.

Tong Mu, Alec Helyar, Johannes Heidecke, Joshua Achiam, Andrea Vallone, Ian Kivlichan, Molly Lin, Alex Beutel, John Schulman, and Lilian Weng. Rule based rewards for language model safety. In *Advances in Neural Information Processing Systems*, 2024.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. InFoBench: Evaluating instruction following ability in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13025–13048. Association for Computational Linguistics, 2024a. URL https://aclanthology.org/2024.findings-acl.772/.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 1504–1518. Association for Computational Linguistics, 2024b. doi: 10.18653/v1/2024.findings-naacl.97. URL https://aclanthology.org/2024.findings-naacl.97/.

Qwen, An Yang, Baosong Yang, Beichen Zhang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

MohammadHossein Rezaei, Robert Vacareanu, Zihao Wang, Clinton Wang, Bing Liu, Yunzhong He, and Afra Feyza Akyürek. Online rubrics elicitation from pairwise comparisons. *arXiv preprint arXiv:2510.07284*, 2025. URL https://arxiv.org/abs/2510.07284.

Thomas L. Saaty. A scaling method for priorities in hierarchical structures. *Journal of Mathematical Psychology*, 15(3):234–281, 1977. doi: 10.1016/0022-2496(77)90033-5. URL https://doi.org/10.1016/0022-2496(77)90033-5.

Thomas L. Saaty. *The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation*. McGraw-Hill, New York, 1980.

Zifei Shan and Fangning Shao. A survey on rubric-guided reinforcement learning for language models. In *Findings of the Association for Computational Linguistics: EMNLP 2026*, 2026. URL https://arxiv.org/abs/2608.27505.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

William F. Shen, Xinchi Qiu, Chenxi Whitehouse, Lisa Alazraki, Shashwat Goel, Francesco Barbieri, Timon Willi, Akhil Mathur, and Ilias Leontiadis. Rethinking rubric generation for improving LLM judge and reward modeling for open-ended tasks. *arXiv preprint arXiv:2602.05125*, 2026. URL https://arxiv.org/abs/2602.05125.

Rickard Stureborg, Dimitris Alikaniotis, and Yoshi Suhara. Large language models are inconsistent and biased evaluators. *arXiv preprint arXiv:2405.01724*, 2024. URL https://arxiv.org/abs/2405.01724.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y. Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. JudgeBench: A benchmark for evaluating LLM-based judges. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.12784.

Vijay Viswanathan, Yanchao Sun, Shuang Ma, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. Checklists are better than reward models for aligning language models. *arXiv preprint arXiv:2507.18624*, 2025. URL https://arxiv.org/abs/2507.18624.

Ran Xu, Tianci Liu, Zihan Dong, Tony Yu, Ilgee Hong, Carl Yang, Linjun Zhang, Tao Zhao, and Haoyu Wang. Alternating reinforcement learning for rubric-based reward modeling in non-verifiable LLM post-training. *arXiv preprint arXiv:2602.01511*, 2026. URL https://arxiv.org/abs/2602.01511.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Li S. Yifei, Allen Chang, Chaitanya Malaviya, and Mark Yatskar. ResearchQA: Evaluating scholarly question answering at scale across 75 fields with survey-mined questions and rubrics. *Transactions of the Association for Computational Linguistics*, 2026. doi: 10.1162/TACL.a.732. URL https://arxiv.org/abs/2509.00496.

Junkai Zhang, Zihao Wang, Lin Gui, Swarnashree Mysore Sathyendra, Jaehwan Jeong, Victor Veitch, Wei Wang, Yunzhong He, Bing Liu, and Lifeng Jin. Chasing the tail: Effective rubric-based reward modeling for large language model post-training. In *International Conference on Learning Representations*, 2026a. URL https://arxiv.org/abs/2509.21500.

Qiyuan Zhang, Junyi Zhou, Yufei Wang, Fuyuan Lyu, Yidong Ming, Can Xu, Qingfeng Sun, Kai Zheng, Peng Kang, Xue Liu, and Chen Ma. RubricBench: Aligning model-generated rubrics with human standards. *arXiv preprint arXiv:2603.01562*, 2026b. URL https://arxiv.org/abs/2603.01562.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. WildChat: 1M ChatGPT interaction logs in the wild. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2405.01470.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 36, 2023. URL https://arxiv.org/abs/2306.05685.

A Related Work

Rubric-Based Evaluation. Rubric-based evaluation is increasingly adopted to evaluate open-ended LLM responses, where response quality is decomposed into explicit criteria verified by LLMs (Hashemi et al., 2024; Li et al., 2025a). Existing works mainly advance the construction, coverage, and downstream use of rubrics. For construction, rubrics are written by experts (Arora et al., 2025; Zhang et al., 2026b), mined from survey articles (Yifei et al., 2026), or generated by LLMs for each user prompt (Cook et al., 2024; Li et al., 2026; Xu et al., 2026). For coverage, rubrics are extended with criteria elicited from pairwise comparisons of responses (Rezaei et al., 2025) or generated to comprehensively capture the evaluation requirements of a user prompt (Chen et al., 2026). For downstream use, evaluation scores are used to rank candidate responses (Li et al., 2026) and serve as rewards for RL beyond verifiable domains (Mu et al., 2024; Gunjal et al., 2026; Viswanathan et al., 2025; Zhang et al., 2026a; Shan & Shao, 2026). Aggregation schemes include both uniform weighting (Qin et al., 2024a; Cook et al., 2024; Lee et al., 2025) and explicit importance weights (Arora et al., 2025; Viswanathan et al., 2025; Li et al., 2026); we focus on how these weights are obtained.

Rubric Weighting. Beyond uniform weighting, some approaches rely on experts to specify criterion weights (Arora et al., 2025), which is costly and difficult to scale. Existing automatic methods can be divided into direct and indirect weighting methods. Direct weighting methods prompt LLMs to assign numerical weights or importance levels to individual criteria or evaluation dimensions (Du et al., 2025; Viswanathan et al., 2025; Gunjal et al., 2026; Zhang et al., 2026a; Ding, 2026; Li et al., 2026). Some of them stabilize the weights by aggregating repeated generations (Du et al., 2025) or assigning coarse importance levels (Gunjal et al., 2026; Zhang et al., 2026a), while still eliciting importance judgments directly. Findings of anchoring effects and inconsistent ratings in text evaluation (Stureborg et al., 2024) motivate examining alternative elicitation schemes. Indirect weighting methods derive the weights from pairwise comparisons or verification results. MAPLE (Jinji et al., 2026) compares criteria pairwise with multiple LLMs, but aggregates the comparisons with a Bradley–Terry model (Bradley & Terry, 1952) without checking their consistency. RRD (Shen et al., 2026) downweights correlated criteria based on the verification results of sampled responses, which reduces repeated credit for overlapping criteria but disregards the importance of the criteria. In contrast, ICW derives the weights from pairwise importance comparisons through the principal eigenvector, and checks and revises the comparisons based on their consistency.

B Theoretical Justification

We provide the proofs of Propositions 1 and 2 and the derivation of Eq. (4) following Saaty (1977).

B.1 Proof of Proposition 1

Proof. If there exist weights $\boldsymbol{w}$ with $w_k > 0$ such that $a_{ij} = w_i/w_j$ for all i, j, then $a_{ij}a_{j\ell} = (w_i/w_j)(w_j/w_\ell) = w_i/w_\ell = a_{i\ell}$ for all i, j, ℓ, so $\mathbf{A}$ is consistent. Conversely, let $\mathbf{A}$ be consistent, and define $w_i = a_{i1}/\sum_{k=1}^{K} a_{k1}$ for $i = 1, \ldots, K$, so that $w_i > 0$ and $\sum_{i=1}^{K} w_i = 1$. For any i and j, consistency and reciprocity give $a_{ij} = a_{i1}a_{1j} = a_{i1}/a_{j1} = w_i/w_j$. If another normalized vector $\boldsymbol{w}'$ also satisfies $a_{ij} = w'_i/w'_j$ for all i, j, then $w'_i/w'_1 = a_{i1} = w_i/w_1$ for all i, so $\boldsymbol{w}'$ is proportional to $\boldsymbol{w}$, and the normalization gives $\boldsymbol{w}' = \boldsymbol{w}$. Moreover, for each i,

$$(\mathbf{A}\boldsymbol{w})_i = \sum_{j=1}^{K} a_{ij}w_j = \sum_{j=1}^{K} \frac{w_i}{w_j}w_j = Kw_i,$$

so $\mathbf{A}\boldsymbol{w} = K\boldsymbol{w}$. Since $\mathbf{A}$ is positive and $\boldsymbol{w}$ is a positive eigenvector, the Perron–Frobenius theorem (Horn & Johnson, 2012) implies that its eigenvalue is the largest eigenvalue of $\mathbf{A}$, i.e., $\lambda_{\max} = K$, and that $\boldsymbol{w}$ is the principal eigenvector. □

B.2 Proof of Proposition 2

Proof. By the Perron–Frobenius theorem (Horn & Johnson, 2012), the principal eigenvector $\boldsymbol{w}$ of $\mathbf{A}$ is positive. Define $\epsilon_{ij} = a_{ij}w_j/w_i > 0$ for all i, j, so that reciprocity gives $\epsilon_{ii} = 1$ and $\epsilon_{ji} = 1/\epsilon_{ij}$.

The i-th row of $\mathbf{A}\boldsymbol{w} = \lambda_{\max}\boldsymbol{w}$ gives $\lambda_{\max} = \sum_{j=1}^{K} \epsilon_{ij}$ for each i. Summing over i yields

$$K\lambda_{\max} = \sum_{i=1}^{K}\sum_{j=1}^{K} \epsilon_{ij} = K + \sum_{1\le i<j\le K} \left(\epsilon_{ij} + \frac{1}{\epsilon_{ij}}\right) \ge K + K(K-1) = K^2,$$

where the inequality follows from $t + 1/t \ge 2$ for $t > 0$, with equality if and only if $t = 1$. Hence, $\lambda_{\max} \ge K$, with equality if and only if $\epsilon_{ij} = 1$, i.e., $a_{ij} = w_i/w_j$, for all i, j. If the equality holds, $\mathbf{A}$ is consistent by Proposition 1; conversely, if $\mathbf{A}$ is consistent, Proposition 1 gives $\lambda_{\max} = K$. Moreover, since there are $K(K-1)/2$ pairs with $i < j$, subtracting K^2 from both sides of $K\lambda_{\max} = K + \sum_{i<j}(\epsilon_{ij} + 1/\epsilon_{ij})$ and dividing by $K(K-1)$ yields Eq. (4). □

C More Experimental Results

In this section, we present supplementary empirical results deferred from the main text. First, we extend the generalization study of Section 4.4 to RubricBench, ResearchQA, and JudgeBench (Section C.1). Second, we extend the sensitivity study of Section 4.5 to other weighting models (Section C.2). Third, we report the token cost of ICW (Section C.3).

C.1 More Generalization Results

In this subsection, we extend the generalization study of Section 4.4 to RubricBench, ResearchQA, and JudgeBench. The results are presented in Tables 6 and 7, and in line with the findings in Section 4.4, we make two observations. **❶ ICW generalizes across verifiers.** With GPT-4.1 as the weighting model, ICW generally outperforms Uniform across the five verifiers on the three benchmarks. This further confirms that the weights of ICW complement the verification results of different verifiers. **❷ ICW generalizes across weighting models.** With Gemini-3.1-Flash-Lite as the verifier, ICW consistently outperforms Uniform with all five weighting models on the three benchmarks. This further confirms that pairwise importance comparison does not rely on a particular weighting model.

Table 6: Results across verifiers on the other benchmarks.

Method	RubricBench		ResearchQA		JudgeBench	
	Acc	RI	Acc	RI	Acc	RI
Verifier: Gemini-3.5-Flash						
Uniform	0.795	–	0.601	–	0.802	–
ICW	**0.804**	**1.16% ↑**	**0.612**	**1.96% ↑**	**0.830**	**3.52% ↑**
Verifier: GPT-4.1-mini						
Uniform	0.760	–	**0.583**	–	0.589	–
ICW	**0.773**	**1.72% ↑**	0.580	0.50% ↓	**0.607**	**3.02% ↑**
Verifier: DeepSeek-V4-Flash						
Uniform	0.788	–	0.592	–	0.704	–
ICW	**0.792**	**0.56% ↑**	**0.604**	**2.01% ↑**	**0.715**	**1.49% ↑**
Verifier: GLM-5.1						
Uniform	0.791	–	0.604	–	0.759	–
ICW	**0.796**	**0.66% ↑**	**0.618**	**2.44% ↑**	**0.775**	**2.12% ↑**
Verifier: Gemini-3.1-Flash-Lite						
Uniform	0.771	–	0.583	–	0.762	–
ICW	**0.795**	**3.11% ↑**	**0.612**	**5.08% ↑**	**0.783**	**2.76% ↑**

Note: "RI" denotes relative improvement over the Uniform baseline.

Table 7: Results across weighting models on the other benchmarks.

Method	RubricBench		ResearchQA		JudgeBench	
	Acc	RI	Acc	RI	Acc	RI
Weighting Model: Gemini-3.5-Flash						
Uniform	0.771	–	0.583	–	0.762	–
ICW	**0.792**	**2.83% ↑**	**0.615**	**5.58% ↑**	**0.782**	**2.65% ↑**
Weighting Model: GPT-4.1-mini						
Uniform	0.771	–	0.583	–	0.762	–
ICW	**0.789**	**2.44% ↑**	**0.607**	**4.07% ↑**	**0.783**	**2.76% ↑**
Weighting Model: DeepSeek-V4-Flash						
Uniform	0.771	–	0.583	–	0.762	–
ICW	**0.795**	**3.11% ↑**	**0.601**	**3.05% ↑**	**0.777**	**2.01% ↑**
Weighting Model: GLM-5.1						
Uniform	0.771	–	0.583	–	0.762	–
ICW	**0.797**	**3.40% ↑**	**0.607**	**4.07% ↑**	**0.782**	**2.65% ↑**
Weighting Model: GPT-4.1						
Uniform	0.771	–	0.583	–	0.762	–
ICW	**0.795**	**3.11% ↑**	**0.612**	**5.08% ↑**	**0.783**	**2.76% ↑**

Note: "RI" denotes relative improvement over the Uniform baseline.

C.2 More Sensitivity Results

In this subsection, we extend the sensitivity study of Section 4.5 to Gemini-3.5-Flash and GLM-5.1 as the weighting models. The results are presented in Figure 4, and in line with the findings in Section 4.5, we make two observations. ❶ **ICW improves over uniform weighting across weighting models.** With both weighting models, ICW outperforms uniform weighting ($\alpha = 1$) on all four benchmarks for any tested $\alpha \geq 2$. This indicates that the benefit of pairwise importance comparison does not depend on a particular weighting model. ❷ **ICW remains robust to the fixed ratio across weighting models.** Once $\alpha \geq 2$, the accuracy remains stable as α further increases for both weighting models. This confirms that the robustness to α observed with GPT-4.1 in Section 4.5 is not tied to a particular weighting model.

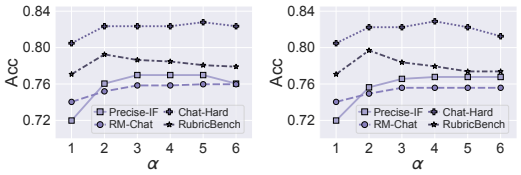

Figure 4: Sensitivity to the fixed ratio α with Gemini-3.5-Flash (left) and GLM-5.1 (right) as the weighting models.

C.3 Cost Analysis

Table 8 reports the token cost of ICW, i.e., the average numbers of input and output tokens consumed by the weighting model per sample.

Table 8: Token cost of ICW with different weighting models.

Weighting Model	Type	RubricBench	ResearchQA	JudgeBench	RM-Chat	Chat-Hard	Precise-IF
GPT-4.1	Input	39.552	152.116	111.641	92.567	107.761	185.989
	Output	1.377	2.957	1.962	1.799	1.872	2.510
Gemini-3.5-Flash	Input	49.799	190.582	127.352	106.721	122.638	231.673
	Output	18.074	41.253	22.049	17.428	17.770	29.697
GLM-5.1	Input	61.862	180.468	113.025	94.347	110.812	197.559
	Output	26.279	53.043	26.773	22.474	23.030	33.580

Note: The numbers are in thousands of tokens per sample, and the output tokens include reasoning tokens.

D Reproduction Details

D.1 Dataset Descriptions

In this section, we document the datasets adopted in our experiments, which serve two purposes: (1) pairwise preference benchmarks for evaluating rubric weighting methods, and (2) training prompts and evaluation benchmarks for downstream RL.

Pairwise Preference Benchmarks. Each benchmark provides user prompts, pairs of candidate responses, and preference labels.

- **RubricBench** (Zhang et al., 2026b)[2]: A benchmark for rubric-guided evaluation that targets hard samples with nuanced input complexity and misleading surface features, where each sample is annotated with expert-written rubrics derived from the instruction.
- **ResearchQA** (Yifei et al., 2026)[3]: A scholarly question answering benchmark distilled from survey articles across research fields, where each query is paired with query-specific rubric items. We use the response pairs with human preference annotations.
- **JudgeBench** (Tan et al., 2025)[4]: A benchmark of challenging response pairs spanning knowledge, reasoning, math, and coding, where the preference labels reflect factual and logical correctness.

[2] https://github.com/planepig/rubricbench
[3] https://huggingface.co/datasets/realliyifei/ResearchQA
[4] https://github.com/ScalerLab/JudgeBench

- **RM-Bench** (Liu et al., 2025)[5]: A reward model benchmark that evaluates sensitivity to subtle content differences and resistance to style biases. We use its Chat subset.
- **RewardBench** (Lambert et al., 2025)[6]: A reward model benchmark spanning chat, reasoning, and safety. We use its Chat-Hard subset, which contains challenging chat pairs with subtle differences.
- **RewardBench 2** (Malik et al., 2026)[7]: A reward model benchmark built on new human prompts. We use its Precise-IF subset, which evaluates whether responses follow precise instruction constraints.

For JudgeBench, RM-Chat, Chat-Hard, and Precise-IF, the rubric sets are generated by the RubricARM-8B-Rubric generator of Rubric-ARM (Xu et al., 2026)[8].

Downstream RL Data and Benchmarks.

- **WildChat** (Zhao et al., 2024): A corpus of real user–ChatGPT conversations. We use its user prompts for RL training, and generate a rubric set for each prompt with Gemini-3.5-Flash.
- **BiGGen Bench** (Kim et al., 2025): A benchmark that evaluates nine capabilities of language models with instance-specific evaluation criteria. We report the score on a 0–100 scale.
- **AlpacaEval 2.0** (Dubois et al., 2024): A benchmark of single-turn instructions, for which we report the win rate (WR) against reference answers and the length-controlled win rate (LC).
- **Arena-Hard** (Li et al., 2025b): A benchmark of challenging user prompts, for which we report the win rate (WR) against a baseline model and the style-controlled win rate (SC) on Arena-Hard-Auto v0.1.

D.2 Implementation Details

In this section, we supplement the implementation details deferred from the main text.

Hyperparameter Settings. We set the maximum number of revision rounds to $T = 1$ in all experiments. The fixed ratio α and the consistency threshold τ are selected for each benchmark, as listed in Table 9.

Table 9: Fixed ratio α and consistency threshold τ for each benchmark.

Hyperparameter	RubricBench	ResearchQA	JudgeBench	RM-Chat	Chat-Hard	Precise-IF
α	2	2	4	3	2	4
τ	0.005	0.005	0.005	0.005	0.005	0.005

Consistency Check Settings. We compute CR in Eq. (5) with the random indices RI_K of Saaty (1980) for $K \leq 15$. For $K > 15$, we use RI_{15}.

Downstream RL Settings. The RL experiments are implemented on top of the `verl` library[9]. We adopt GRPO (Shao et al., 2024) as the policy optimization algorithm, and initialize the policy models from Qwen2.5-3B-Instruct, Qwen2.5-7B-Instruct (Qwen et al., 2024), and Qwen3-4B (Yang et al., 2025), with the thinking mode enabled for Qwen3-4B. User prompts from WildChat (Zhao et al., 2024) are used for policy training. All methods share the same rubric sets, and the weights are obtained with GPT-4.1 as the weighting model. The reward of each response is its evaluation score in Eq. (1) with GPT-4.1-mini as the verifier. The training is configured with a batch size of 128, 8 rollouts per prompt, and a sampling temperature of 1.0. The policy models are optimized with a learning rate of 1×10^{-6} and a KL coefficient of 1×10^{-3} for 3 epochs. Prompt and response are capped at 2,048 and 9,216 tokens, respectively. For evaluation, GPT-4.1 is used as the judge on all benchmarks.

[5] https://huggingface.co/datasets/THU-KEG/RM-Bench
[6] https://huggingface.co/datasets/allenai/reward-bench
[7] https://huggingface.co/datasets/allenai/reward-bench-2
[8] https://huggingface.co/OpenRubrics/RubricARM-8B-Rubric
[9] https://github.com/volcengine/verl.

Prompt Templates. Prompts D.1–D.3 present the prompts used in our experiments. The weighting model compares each criterion pair with Prompt D.1, where {revision_feedback} is empty in the initial comparisons and is replaced by Prompt D.2 during revision. The verifier checks each criterion for each response with Prompt D.3, for both the pairwise preference benchmarks and downstream RL.

Prompt D.1: Pairwise importance comparison prompt of the weighting model.

System message: You compare the importance of two rubric criteria for one user prompt. Importance means scoring weight: when grading candidate answers to this prompt, failing a more important criterion must cost more credit. Use only the user prompt and the fixed rubric set. Never guess at candidate answers, human preferences, or verifier outcomes.

User message:

User prompt:
{prompt}

Full rubric set (fixed context for this sample):
{full_rubric_set}

Compare these two rubrics:
first ({first_rubric_id}): {first_criterion}
second ({second_rubric_id}): {second_criterion}

Which highlighted rubric should carry more weight when grading answers to this prompt?

General rules:
- First list what the prompt itself fixes: the requested result and every explicit must, must-not, count, format, or procedural rule. A rubric tied to any of them outranks generic quality.
- Rank two rubrics only when the prompt gives a reason; never build a preference from plausible reasoning alone. If neither rubric is favored by the prompt, answer "neutral".
- Judge the concrete behavior each rubric uniquely requires, not wording such as detailed, complete, robust, or professional.
- Use the full rubric set: a duplicate, vague umbrella, or whole-task restatement is less important than a specific unique requirement.
- For a negatively phrased rubric, judge the value of avoiding that defect; do not reverse its meaning.
- First decide which rubric_id wins; your answer must not change if the two rubrics were displayed in the opposite order.

{revision_feedback}

Return JSON only:
{"more_important": "first" | "second" | "neutral", "reason": "<= 25 words"}

Prompt D.2: Revision prompt of the weighting model.

Consistency review: the initial judgments for every rubric pair follow. Each forward/reverse entry preserves its display order, winner_rubric_id, and original reason. A null winner means that call answered neutral. The status distinguishes:
- agreed_winner: both display orders support the same rubric_id;
- both_neutral: both calls report no supported importance advantage;
- one_order_neutral: only one call is neutral;
- opposite_winners: the two calls support different rubric_ids.
The last two statuses indicate order instability. Their null collapsed winner does not establish that the rubrics are equally important. Even both_neutral can reflect insufficient evidence rather than an established equality.

Initial comparison evidence (fallible prior judgments, not ground truth):
{previous_round_history}

Re-judge the highlighted pair using the user prompt and complete rubric set, checking both previous reasons against the prompt. Use the other pairs as context, not as votes or authoritative evidence. Keep a supported judgment; revise it when its reason is unsupported, misreads a requirement, or overlooks a relevant requirement. Explain the concrete prompt or rubric evidence for your judgment. A request for review does not imply that the highlighted pair's initial judgment was wrong. Seek coherent importance judg-

ments, but do not invent a preference merely to remove a conflict. Answer "neutral" when neither rubric is supported as more important. Return a judgment only for the highlighted pair.

Prompt D.3: Prompt of the verifier.

System message: You are a strict rubric verifier. Given one user prompt, one response, and ONE evaluation criterion, decide whether the response satisfies that criterion. Judge only this local criterion, not overall response quality, and never compare against another response. Mark satisfied=1 only when the response clearly and completely satisfies the criterion; partial satisfaction is 0. Return strict JSON only.

User message:

Original user prompt:
{prompt}

Response to verify:
{response}

Single criterion:
{criterion}

Verification rules:
- Judge only the criterion's stated local boundary. Do not fail it merely because another part of the overall task is weak or missing.
- Preserve exact quantifiers: exactly, at least, at most, only, every, and each are not interchangeable.
- If the criterion forbids or says to avoid something, satisfied=1 only when the prohibited condition is absent.
- Judge meaning and response evidence, not keyword overlap.
- The reason must be one concise final justification. The satisfied bit and reason must agree.

Return exactly:
{"satisfied": 1, "reason": "short evidence-based reason"}