FROM CASES TO CRITERIA: REUSING EVALUATION KNOWLEDGE FOR TASK-SPECIFIC RUBRIC GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Query-specific rubrics make LLM evaluation more fine-grained and transparent, and generating high-quality rubrics requires both identifying the target query's evaluation scope and specifying concrete checks. Historical rubrics provide reusable evaluation experiences, yet existing approaches reuse either complete query–rubric cases or aggregated criteria and dimensions without explicitly connecting recurring evaluation requirements to their criterion-level evidence and source-query context. This can preserve irrelevant task details or lose the broader evaluation structure. In this paper, we introduce *RubricDuo*, which organizes historical query–rubric pairs into a Rubric Knowledge Graph linking evaluation dimensions, shared concepts, local concepts, criteria, and queries, where reusable evaluation requirements across queries are abstracted by shared concepts. Given a target query, *RubricDuo* first combines concepts relevant to the target query with retrieved historical cases to draft a rubric with broad coverage. It then revisits each draft criterion, retrieves concept-linked historical criteria and their source queries, and adapts applicable evidence to refine the criterion. Experiments on RubricBench and HealthBench demonstrate that *RubricDuo* achieves the overall best performance among the evaluated automatic methods in both alignment with human-authored rubrics and downstream LLM judging.

1 INTRODUCTION

Large language models (LLMs) are increasingly evaluated by other LLMs acting as judges (Liu et al., 2023; Zheng et al., 2023), especially for complex, open-ended tasks. However, a judge that outputs only a score (Lee et al., 2024) or a preference (Wang et al., 2024) reveals little about why one response is better than another. Rubric-based evaluation makes this reasoning explicit by decomposing the requirements of a query into criteria that can be checked individually (Hashemi et al., 2024; Kim et al., 2024; Cook et al., 2024). Expert-written rubrics now underpin benchmarks such as HealthBench (Arora et al., 2025) and will also serve as reward signals for training LLMs in domains without verifiable answers (Gunjal et al., 2026). Since writing a rubric for every query requires substantial expert effort, a growing line of work seeks to generate rubrics automatically.

A central challenge in automatic rubric generation is deciding what criteria should check. On RubricBench (Zhou et al., 2026), replacing model-generated rubrics with human-written ones raises judge accuracy by about 27 points, and the gap persists as more rubrics are sampled. Historical human rubrics are a natural source of the missing knowledge because they record which information experts checked for related queries. Recent methods therefore reuse this historical knowledge, retrieving query–rubric pairs similar to the target query as demonstrations (Dhole & Agichtein, 2026) or by distilling historical criteria into pools of evaluation dimensions or criteria (Mao et al., 2026; Jia et al., 2026). Nevertheless, such reuse fails to reliably improve the quality of the generated rubrics. In our experiments (§4), contextualizing the generator with retrieved similar cases fails to improve alignment with human rubrics compared to generating from the query alone in three of five settings.

We argue that this unreliability stems from the unit at which rubric knowledge is represented and reused. What transfers across queries is usually not a whole rubric or the literal wording of a criterion, but the evaluation requirement that a criterion instantiates at the conceptual level. Consider the

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.

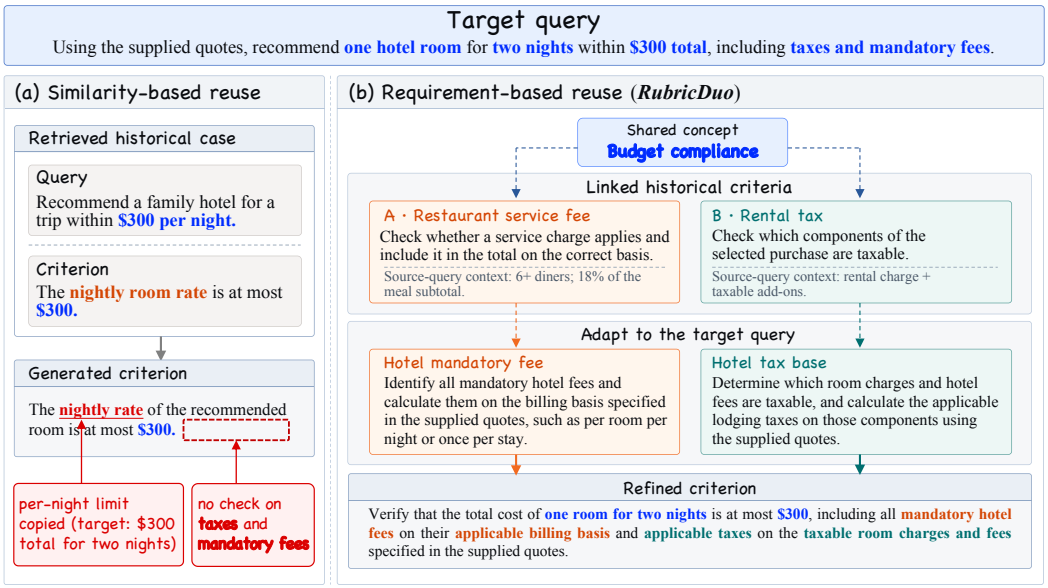


Figure 1: Similarity- and requirement-based rubric reuse. (a) A similar hotel case carries over an incorrect nightly limit and misses taxes and mandatory fees. (b) A shared concept links criteria from different domains to their source query contexts; applicable checks are adapted to verify the $300 two-night total, including fees and taxes.

hotel recommendation query in Fig. 1(a), which asks for a hotel room for two nights within $300 in total, including taxes and mandatory fees. Similarity-based reuse favors other hotel queries, such as a request for a family hotel within $300 per night, whose rubric bounds the nightly rate rather than the total and says nothing about mandatory fees. The missing check appears instead in the rubric of a restaurant reservation, which requires the total to include a service charge for large parties, and a car-rental rubric adds a check on which charges are taxable. These queries differ in topic, but their criteria express the same underlying requirement, i.e., the reported total must include every applicable charge. Beyond similar case retrieval, effective reuse therefore entails a well-refined abstraction and organization of recurring evaluation requirements across historical cases with various topics. First, these evaluation requirements should be abstracted from historical rubrics regarding different evaluation dimensions. Second, historical criteria and queries should be indexed with these evaluation requirements, enabling the generator to retrieve knowledge based on the requirements of target queries while retaining the source context needed to adapt that knowledge. Existing methods do not provide such a joint knowledge abstraction and organization. Case retrieval preserves source context but operates at the query level, while pools of dimensions or aggregated criteria capture recurring patterns without explicitly organizing how an abstract requirement connects back to the historical evidence that instantiates it.

To this end, we propose *RubricDuo* in this paper, which reuses historical rubrics through their shared evaluation requirements. In the offline phase, *RubricDuo* organizes historical query–rubric pairs into a Rubric Knowledge Graph. The graph abstracts each criterion into the requirements it checks, groups recurring requirements into shared evaluation concepts such as '*budget compliance*', and links every concept back to its source criteria and queries. In the online phase, *RubricDuo* generates a rubric in two steps. In the first step, it selects the shared concepts relevant to the target query, retrieves similar historical cases, and drafts a rubric that covers the overall evaluation scope. Similar cases remain useful at this stage because they show how a set of criteria jointly covers a problem, while the selected concepts indicate which requirements the target query raises. In the second step, *RubricDuo* revisits each draft criterion, retrieves historical criteria linked to the same concept, and adapts only the details that apply to the target query. Because candidates are filtered by concept rather than by query similarity, this evidence can come from tasks unrelated in topic, as shown in Fig. 1(b).

We evaluate *RubricDuo* on RubricBench (Zhou et al., 2026) and HealthBench (Arora et al., 2025), using both alignment with held-out human rubrics and downstream LLM-judge accuracy. Across five settings covering three rubric generators and cross-benchmark transfer from RubricBench to HealthBench, *RubricDuo* achieves the highest alignment F1 score in all five settings and the highest judging accuracy in four. On RubricBench, it improves judging accuracy from 45.51% with query-only rubrics to 55.56%, closing 37% of the gap to human-written rubrics. Controlled comparisons further show that the gain comes from how historical knowledge is organized and reused: replacing graph-guided retrieval with flat dense or hybrid retrieval reduces accuracy by more than six points, while concept-guided drafting alone already outperforms all baselines.

2 Related Work

Rubric Generation from Target Queries. Query-conditioned methods generate evaluation criteria directly from the requirements of a target query. Given the target query and its associated topic, Wu et al. (2025) generates instance-specific rubrics. TICK (Cook et al., 2024) and RocketEval (Wei et al., 2025) identify individual query requirements and express them as respective rubric items. Qworld (Su et al., 2026) decomposes a query into scenarios and evaluation perspectives, then develops fine-grained criteria through recursive tree-based exploration. These methods improve how generators identify and express evaluation requirements, while our work focuses on how historical evaluation knowledge can be organized and supplied to the generator, with explicit connections between reusable concepts and concrete query-rubric instances.

Rubric Construction from Historical Experience. Historical evaluation experience can be reused for new queries through complete cases, shared candidate pools, or criteria aggregated across queries. RubricRAG (Dhole & Agichtein, 2026) retrieves semantically similar historical query–rubric pairs as few-shot demonstrations to guide rubric generation for a target query. CaRG (Mao et al., 2026) induces a reusable evaluation dimension pool from historical queries and selects pertinent dimensions to guide generation. Auto-Rubric (Xie et al., 2026) and SibylSense (Xu et al., 2026) derive criteria from comparisons between preferred and dispreferred responses and aggregate these criteria across queries for repurposing on new queries. Specifically, Auto-Rubric distills the aggregated criteria into a compact, query-agnostic coreset, while SibylSense organizes the criteria by category. Prior studies (Qiao et al., 2026; Wang et al., 2026) demonstrate different ways to reuse evaluation experience, but only from either the query-side or the criterion-side. In contrast, *RubricDuo* connects the two through shared evaluation concepts as the provenance of cross-query evaluation requirements. This provenance allows complete historical cases to guide rubric coverage while enabling individual historical criteria to provide concrete evidence.

3 *RubricDuo*: Methodology

Problem Setup. Let $\mathcal{H} = \{(q_i, R_i)\}_{i=1}^{N}$ denote a collection of historical queries and their associated rubric, where the rubric for query q_i contains a set of criteria, i.e., $R_i = \{r_{ij}\}_{j=1}^{m_i}$, for evaluating a response to q_i across m_i facets. Throughout this paper, *rubric* refers to the complete set of criteria for a query, while *criterion* refers to an individual item in the rubric. Given a new query q, our goal is to generate a rubric $\widehat{\mathcal{R}}_q$ that captures its evaluation requirements comprehensively and precisely.

Overview. Fig. 2 delineates an overview of *RubricDuo*, which comprises two phases: offline multi-level *Rubric Knowledge Graph* (RKG) construction (§3.1) and online concept-guided rubric generation (§3.2). In the offline phase, *RubricDuo* constructs an RKG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ from historical evaluation cases $\mathcal{H}$. The graph connects original queries and criteria with derived local concepts, shared concepts, and evaluation dimensions in a level-by-level fashion. More specifically, to induce concepts, *RubricDuo* specifies six different general evaluation dimensions from criteria, including Domain, Task, Response Protocol, Constraint, General Quality, and Safety & Policy. The constructed multi-level RKG preserves provenance at multiple granularities, linking transferable evaluation dimensions and concepts to concrete historical criteria and queries. In the online phase, *RubricDuo* generates a rubric for a new query q using a two-step, concept-guided strategy. The first step performs coarse-grained drafting to cover general evaluation aspects of q, and then the second step performs fine-grained refinement to make each criterion concrete and consistent. Query-level

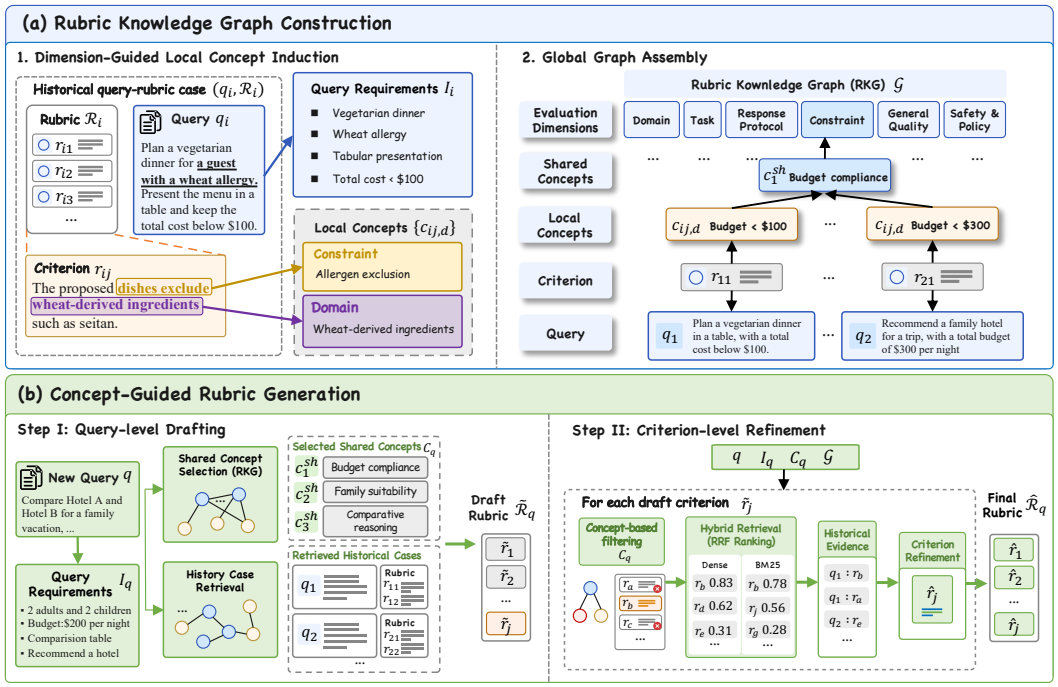


Figure 2: The overview of *RubricDuo*: (a) The offline phase constructs a Rubric Knowledge Graph guided by six dimension; (b) the online phase generates rubric for a target query via query-level drafting and criterion-level refinement.

drafting retrieves similar queries and their rubrics and uses them as demonstrations to produce an initial rubric by prompting an LLM generator. Subsequently, criterion-level refinement revisits each criterion in the initial rubric. For each criterion, *RubricDuo* retrieves a set of historical criteria in $\mathcal{G}$ by concept-guided filtering and reranking, and then refers them as detailed evidence to refine the draft criterion, yielding the final rubric.

3.1 Rubric Knowledge Graph Construction

RubricDuo represents the Rubric Knowledge Graph (RKG) as a multi-level heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The graph contains five types of nodes, i.e., $\mathcal{V} = \mathcal{V}_Q \cup \mathcal{V}_R \cup \mathcal{V}_C^{\text{loc}} \cup \mathcal{V}_C^{\text{sh}} \cup \mathcal{V}_D$, where $\mathcal{V}_Q$ and $\mathcal{V}_R$ denote the historical queries and individual criteria from $\mathcal{H}$, respectively. $\mathcal{V}_C^{\text{loc}}$ and $\mathcal{V}_C^{\text{sh}}$ denote the local and shared concepts induced from $\mathcal{H}$, while $\mathcal{V}_D$ denotes six general evaluation dimensions. Intuitively, local concepts $\mathcal{V}_C^{\text{loc}}$ capture distinct evaluation requirements delivered by a criterion, while shared concepts $\mathcal{V}_C^{\text{sh}}$ represent semantic-level evaluation requirements that recur across multiple criteria and are organized by their corresponding evaluation dimensions $\mathcal{V}_D$.

3.1.1 Dimension-Guided Local Concept Extraction

RubricDuo defines six general evaluation dimensions $\mathcal{V}_D$ to guide local concept extraction and shared concept induction, enabling fine-grained reuse of historical experience. These dimensions distinguish six evaluation requirements within a criterion: (1) *Domain* for domain-specific factual or substantive content; (2) *Task* for the objective or operation that the response must perform; (3) *Response Protocol* for requirements concerning format, organization, or presentation; (4) *Constraint* for explicit limits, conditions, or resources; (5) *General Quality* for cross-task qualities such as clarity, coherence, and completeness; (6) *Safety & Policy* for safety, ethical, legal, or policy-related requirements.

For each historical query q_i, we first extract its intent and requirements, i.e., $I_i := \text{IntentExtract}(q_i)$. Given the source query q_i and its extracted requirements I_i, we identify the dimensions presented within each criterion $r_{ij} \in \mathcal{R}_i$ as $D_{ij} := \text{DimensionIdentify}(r_{ij} \mid$

$q_i, I_i), D_{ij} \subseteq \mathcal{V}_D$, where D_{ij} is a subset of the six dimensions corresponding to the general aspects evaluated by criterion r_{ij}. The query intent extraction and dimension identification are performed by prompting an LLM with the corresponding contexts.

For each identified dimension $d \in D_{ij}$, we then extract a local concept $c_{ij,d}$ from r_{ij} that captures a concrete evaluation aspect within r_{ij}. The extraction is likewise performed through LLM prompting, conditioned on the query q_i and its intent I_i, criterion r_{ij}, and a dimension d, i.e., $c_{ij,d} := \mathrm{ConcepetExtract}(r_{ij} \mid q_i, I_i, d)$. The local concepts extracted from all queries and their criteria form the local concept entities $\mathcal{V}_C^{\mathrm{loc}}$.

For illustration, consider the criterion in Fig. 1: '*The dinner total is at most $180, including the mandatory 18% service charge on the pre-tax meal subtotal because the party meets the six-person threshold.*' This criterion evaluates both the spending constraint and the restaurant-specific fee rule. Under *Constraint*, it can yield local concepts such as *dinner spending limit*; under *Domain*, it can yield *restaurant service-fee rule*. The dimensions thus guide the induction of distinct local concepts from the same criterion, while each concept retains its specific evaluation focus and source context.

3.1.2 Shared Concept Induction & Graph Assembly

Shared Concept Induction. *RubricDuo* aggregates semantically relevant local concepts $\mathcal{V}_C^{\mathrm{loc}}$ and induces a set of shared concepts $\mathcal{V}_C^{\mathrm{sh}}$, aiming to enable the explicit reuse of evaluation requirements within criteria across queries. Specifically, we group all extracted local concepts $\mathcal{V}_C^{\mathrm{loc}}$ from $\mathcal{H}$ by their dimension d, and prompt an LLM for each group to induce shared concepts for each dimension-specific group. When a group of local concepts exceeds the effective context window of the LLM, we divide the group into batches and process one batch per prompt. We maintain the shared concepts induced across batches and dimensions by merging semantically equivalent concepts and incrementally adding new concepts, yielding the final shared concept set $\mathcal{V}_C^{\mathrm{sh}}$.

For example, one criterion requires a dinner plan to cost less than $100, while another requires a travel itinerary to stay within a $200 budget. The local concepts induced from these criteria under Constraint both concern staying within a specified budget. We group them under the shared concept '*budget compliance*', while retaining the different tasks and budget limits in their source criteria.

Graph Assembly. With the six dimensions $\mathcal{V}_D$, the extracted local concepts $\mathcal{V}_C^{\mathrm{loc}}$, and the induced shared concepts $\mathcal{V}_C^{\mathrm{sh}}$, *RubricDuo* assembles these entities together with the original queries $\mathcal{V}_Q$ and rubrics $\mathcal{V}_R$ into a multi-level RKG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, as shown in Fig. 2. These five levels of entities are connected by four types of edges, i.e., $\mathcal{E} = \mathcal{E}_{q\to r} \cup \mathcal{E}_{r\to \mathrm{loc}} \cup \mathcal{E}_{\mathrm{loc}\to \mathrm{sh}} \cup \mathcal{E}_{\mathrm{sh}\to d}$. Here, $\mathcal{E}_{q\to r}$ links each historical query to its original criterion; $\mathcal{E}_{r\to \mathrm{loc}}$ links each criterion to its extracted local concepts; $\mathcal{E}_{\mathrm{loc}\to \mathrm{sh}}$ links each local concept to its corresponding shared concept; and $\mathcal{E}_{\mathrm{sh}\to d}$ links each shared concept to its evaluation dimension. These connections allow each shared concept to provide access to both individual criteria and their complete historical cases.

3.2 Concept-Guided Rubric Generation

Given a new query q, *RubricDuo* generates a query-specific rubric $\widehat{\mathcal{R}}_q$ by retrieving and exploiting relevant historical evaluation cases and shared concepts in the RKG $\mathcal{G}$. Generation proceeds in two steps. First, coarse-grained generation drafts an initial rubric $\widetilde{\mathcal{R}}_q$ that covers the general evaluation aspects of q and similar historical queries. Fine-grained refinement then revises each criterion in $\widetilde{\mathcal{R}}_q$ individually to produce concrete and consistent criteria as the final rubric $\widehat{\mathcal{R}}_q$.

Step-I: Query-level Drafting. This step first extracts the intent and requirements of the target query q, i.e., $I_q := \mathrm{IntentExtract}(q)$. Subsequently, an LLM is prompted to select shared concepts in $\mathcal{V}_C^{\mathrm{sh}}$ that are relevant to the query intent, such as $C_q := \mathrm{ConceptIdentify}(q, I_q; \mathcal{G}), C_q \subseteq \mathcal{V}_C^{\mathrm{sh}}$, which identifies a subset of shared concepts $C_q = \{c_1^{\mathrm{sh}}, c_2^{\mathrm{sh}}, \cdots\}$ that match the intents of q. Meanwhile, *RubricDuo* retrieves historical query-rubric cases whose queries are semantically and lexically similar to the target query q. Historical queries are ranked using both dense semantic retrieval and BM25 retrieval, and the two rankings are combined using Reciprocal Rank Fusion (Cormack et al., 2009). The top-K retrieved cases, $\mathcal{H}_q = \{(q_k, \mathcal{R}_k)\}_{k=1}^{K}$, demonstrate how multiple criteria

jointly cover an evaluation problem, while the selected shared concepts C_q indicate which evaluation requirements are relevant to the target query. We provide the target query q, its extracted intent I_q, the selected concepts C_q, and the retrieved cases $\mathcal{H}_q$ for the rubric generator to draft the initial rubric:

$$\widetilde{\mathcal{R}}_q := \mathrm{Draft}(q, I_q, C_q, \mathcal{H}_q). \tag{1}$$

The resulting draft $\widetilde{\mathcal{R}}_q$ is intended to provide broad coverage before criterion-specific evidence is introduced.

Step-II: Criterion-level Refinement. For each draft criterion $\widetilde{r}_j \in \widetilde{\mathcal{R}}_q$, *RubricDuo* first identifies what the criterion evaluates and maps it to the relevant shared concepts in C_q. These concepts are used to filter historical criteria in the graph $\mathcal{G}$, and those aligned with the relevant concepts are retained as a candidate reference pool for criterion $\widetilde{r}_j$. Within this candidate pool, hybrid retrieval combines dense and BM25 rankings using Reciprocal Rank Fusion to retrieve criteria with similar terms, conditions, or format requirements to $\widetilde{r}_j$. The resulting criteria, along with their source queries, constitute the specific reference evidence E_j for $\widetilde{r}_j$. The rubric generator evaluates this evidence E_j in the context of the target query q and its intent I_q to refine the draft criterion:

$$\widehat{r}_j := \mathrm{Refine}(\widetilde{r}_j; q, I_q, E_j) \quad \forall \widetilde{r}_j \in \widetilde{\mathcal{R}}_q. \tag{2}$$

The final rubric is obtained by refining every draft criterion and assembling the resulting criteria $\widehat{R}_q$.

4 Experiment

We give the experimental setup (§4.1) and evaluate *RubricDuo* by investigating the following research questions. **RQ-1**: Does *RubricDuo* generate rubrics that better recover the evaluation requirements from human annotators and improve the downstream LLM judges? (§4.2) **RQ-2**: Can an RKG constructed from historical query-rubric pairs support knowledge transfer across benchmarks for rubric generation? (§4.3) **RQ-3**: Does organizing historical query-rubric observations as a graph enable more effective rubric construction than flat retrieval? (§4.4) **RQ-4**: Is the two-step generation strategy of *RubricDuo* effective? (§4.5) **RQ-5**: Does structured historical evaluation knowledge remain effective across rubric generators of different capabilities? (§4.6)

4.1 Experimental Setup

Benchmarks. We evaluate rubric generation on RubricBench (Zhou et al., 2026) and HealthBench (Arora et al., 2025). RubricBench contains 1,147 queries with human-authored atomic rubrics, paired responses, and human preference labels, spanning chat, instruction following, STEM, coding, and safety tasks; we use 917 query-rubric pairs as historical observations and reserve 230 examples for held-out evaluation. HealthBench is curated by OpenAI in collaboration with 262 physicians across 60 countries, pairing realistic healthcare conversations with human-authored rubrics. We use 1,000 query-rubric pairs as historical observations and evaluate on 255 held-out queries using the response pairs constructed by *RubricRAG* (Dhole & Agichtein, 2026).

Baselines. We compare *RubricDuo* with five baselines for automatic rubric construction and include two reference conditions. *No-Rubric* evaluates candidate responses without an explicit rubric, while *Human-Rubric* provides the held-out human-authored rubric to the same judge as an oracle reference. Among the automatic baselines, *Direct-Q* generates rubrics from the query alone; *Random-FS* conditions generation on randomly sampled historical query-rubric pairs; *RubricRAG* uses semantically similar historical query-rubric pairs as generation exemplars; *Rubric-Ret* uses semantically similar historical criteria as generation exemplars; and *CaRG* constructs rubrics from a retrieved pool of evaluation dimensions.

Weight Assignment. RubricBench uses unweighted criteria, whereas HealthBench assigns signed point values. Because some methods, such as *CaRG*, generate criteria without weights, we use a shared LLM-based weight assigner (Qwen3.7-flash) on HealthBench. The assigner takes the query and generated rubrics as input and assigns a signed point value to each criterion, ensuring a consistent weighting protocol across methods. In Appendix A.4, we further examine the effect of weighting strategy by comparing shared and native weights for *RubricRAG* and *RubricDuo* on the same generated criteria, using G>B accuracy.

Table 1: Main results on RubricBench and HealthBench. Values are percentages: mean (sample SD) over three runs.

Method	RubricBench				HealthBench			
	Alignment			Utility	Alignment			Utility
	P ↑	R ↑	F1 ↑	Acc. ↑	P ↑	R ↑	F1 ↑	G>B Acc. ↑
No-Rubric	–	–	–	46.96 (1.33)	–	–	–	–
Human-Rubric	–	–	–	72.61 (1.17)	–	–	–	55.56 (1.48)
Direct-Q	48.47 (1.09)	51.40 (0.68)	47.70 (0.86)	45.51 (1.53)	52.27 (0.91)	31.36 (0.53)	35.25 (0.53)	39.35 (0.82)
Random-FS	45.93 (1.48)	47.75 (1.67)	45.15 (1.47)	49.57 (0.44)	62.24 (0.15)	37.70 (0.28)	42.45 (0.19)	46.21 (1.82)
RubricRAG	42.52 (1.92)	48.82 (1.54)	44.03 (1.77)	49.28 (0.66)	62.43 (0.92)	40.20 (0.94)	44.77 (0.99)	47.62 (2.06)
Rubric-Ret	45.56 (1.29)	53.34 (1.21)	48.92 (1.32)	49.13 (1.15)	62.84 (0.95)	38.06 (1.03)	44.11 (0.86)	48.37 (1.34)
CaRG	18.42 (0.80)	50.31 (1.61)	25.77 (1.06)	52.22 (0.44)	22.96 (0.20)	**55.81** (0.40)	30.88 (0.17)	44.71 (0.78)
RubricDuo	**48.48** (0.62)	**59.07** (0.50)	**50.72** (0.57)	**55.56** (1.94)	**62.98** (1.46)	41.33 (1.27)	**46.89** (1.07)	**49.33** (1.81)

Metrics. Following prior work (Zhou et al., 2026; Dhole & Agichtein, 2026), we evaluate Rubric Alignment using Rubric Precision, Rubric Recall, and Structure F1 against held-out human-authored rubrics. For downstream utility, we report pairwise accuracy (Acc.) on RubricBench, the proportion of cases where the judge selects the human-preferred response; and good-over-bad accuracy (G>B Acc.) on HealthBench, the proportion of pairs where the good response scores strictly higher than the bad response under the generated rubric and its criterion weights. All methods use the same generator model and fixed LLM judge, history-based methods also share the same historical corpus. Further metric details are provided in Appendix A.1.

Models and Settings. In the main experiments, all methods use Qwen3.7-flash for rubric generation, and Qwen3.7-flash serves as the downstream judge and weight assigner. History-based methods share the same historical corpus. We use Qwen3-enbedding-4b for dense retrieval. We repeat each automatic method 3 times and report the mean and sample standard deviation. Detailed evaluation settings are provided in the Appendix A.2.

4.2 Main Results

Rubric Alignment. Table 1 shows that *RubricDuo* achieves the best rubric alignment among all methods on RubricBench, with precision of 48.48%, recall of 59.07%, and F1 of 50.72%. The higher recall indicates that *RubricDuo* covers more requirements in the human-authored rubrics, while the higher precision indicates that a larger proportion of its generated criteria match the reference criteria. Moreover, retrieving historical examples alone does not consistently improve alignment: both *Random-FS* and *RubricRAG* obtain lower F1 than *Direct-Q* on RubricBench. This suggests that access to historical rubrics is insufficient without effective selection and adaptation to the current query. On HealthBench, *RubricDuo* also achieves the highest alignment scores, with structure F1 of 46.89%.

Downstream Utility. On RubricBench, *RubricDuo* achieves the highest accuracy among all methods at 55.56%, improving upon *RubricRAG* by 6.28 percentage points and *No-Rubric* by 8.60 percentage points. On HealthBench, *RubricDuo* also achieves the highest G>B accuracy among all methods at 49.33%. However, gaps to *Human-Rubric* remain: 17.05 percentage points on RubricBench and 6.23 percentage points on HealthBench.

We also observe that *CaRG* achieves lower F1 than *Direct-Q* but higher downstream accuracy on both benchmarks. This indicates that stronger agreement with human-authored rubrics does not necessarily translate into better downstream performance, as criteria useful for distinguishing response quality may extend beyond those included in the reference rubric. The improvements of *RubricDuo* in both F1 and accuracy across the two benchmarks suggest that its generated rubrics support both the recovery of human-authored evaluation requirements and effective downstream response evaluation.

Table 2: Cross-benchmark transfer from RubricBench History to HealthBench.

Method	**P** ↑	**R** ↑	**F1** ↑	**G>B Acc.** ↑
Direct-Q	52.27 (0.91)	31.36 (0.53)	35.25 (0.53)	39.35 (0.82)
Random-FS	54.82 (1.01)	30.89 (0.49)	35.56 (0.58)	41.49 (1.74)
RubricRAG	54.19 (1.28)	30.34 (0.78)	35.00 (0.72)	39.75 (3.05)
Rubric-Ret	52.59 (0.80)	30.33 (0.02)	34.57 (0.34)	42.61 (1.58)
CaRG	20.05 (0.43)	**42.21** (0.60)	24.70 (0.53)	38.48 (2.81)
RubricDuo	**57.45** (0.61)	30.55 (0.49)	**37.06** (0.60)	**45.16** (1.24)

Table 3: Comparison of *Flat-Dense*, *Flat-Hybrid*, and *Graph-Guided* retrieval within the same draft-then-refine framework on RubricBench.

Retrieval	P ↑	R ↑	F1 ↑	Acc. ↑
Flat-Dense	46.89 (0.46)	56.41 (0.78)	49.41 (0.54)	48.84 (1.65)
Flat-Hybrid	47.39 (0.61)	57.03 (1.02)	49.97 (0.72)	49.13 (1.15)
Graph-Guided	**48.48** (0.62)	**59.07** (0.50)	**50.72** (0.57)	**55.56** (1.94)

4.3 Cross-Benchmark Transfer

Table 2 evaluates whether historical evaluation knowledge organized in a rubric graph can transfer across benchmarks and support rubric construction under a shift in task distribution. We construct the graph from RubricBench history and apply it to HealthBench queries without using HealthBench historical cases. All methods show limited alignment with human-authored rubrics, achieving Structural F1 scores of approximately 36%. This is because experts across benchmarks may follow different professional standards, RubricBench covers a broad range of general-purpose tasks, whereas HealthBench focuses on specialized healthcare scenarios that require more clinically specific evaluation criteria. Consequently, directly using RubricBench examples to guide rubric generation for HealthBench queries provides limited benefit and may even reduce downstream accuracy. *CaRG* achieves the highest Structural Recall against the human-authored rubrics. This may be explained by a mismatch in criterion polarity across the two benchmarks: RubricBench contains relatively few penalty criteria, whereas HealthBench includes many criteria that penalize unsafe or incorrect responses. Methods that rely primarily on historical RubricBench examples therefore tend to generate predominantly positive criteria. In contrast, *CaRG* generates both rewarding and penalizing criteria around each retrieved evaluation dimension, enabling broader coverage of the HealthBench reference rubrics. *RubricDuo* achieves the highest Structure F1 and accuracy, suggesting that structured knowledge reuse remains valuable for evaluating responses across benchmarks, even when alignment with target expert rubrics is limited.

4.4 Graph Organization versus Flat Retrieval

We examine whether graph-guided retrieval improves rubric refinement over flat dense retrieval on RubricBench. All variants follow the same draft-then-refine framework. (1) *Flat-Dense* first retrieves complete historical query-rubric cases based on query-level dense similarity and uses them to generate an initial draft. For each draft criterion, it then retrieves semantically similar historical criteria to support refinement. (2) *Flat-Hybrid* follows the same two-stage pipeline as *Flat-Dense*, but combines dense and BM25 retrieval using reciprocal rank fusion (RRF) to integrate semantic and lexical matching signals. (3) *Graph-Guided* is our method, *RubricDuo*. It follows the same two-stage framework and uses graph-guided retrieval to provide structured historical evidence for rubric refinement. As shown in Table 3, *Graph-Guided* method achieves an F1 of 50.72% and an accuracy of 55.56%, exceeding *Flat-Dense* and *Flat-Hybrid*. This is because the rubric knowledge graph connects historical cases through shared concepts and evaluation dimensions, enabling the refiner to combine complementary evidence when adapting the rubric to the current query. *Flat-Hybrid* achieves stronger performance than *Flat-Dense*. This improvement may arise because BM25 complements semantic retrieval with exact lexical matching, allowing the model to retrieve criteria

Table 4: Two-step draft-refine ablation on RubricBench.

Step	Precision	Recall	F1	Accuracy
RubricDuo (Draft)	47.33 (1.24)	57.56 (0.74)	49.77 (0.81)	53.63 (0.27)
RubricDuo (Refined)	48.48 (0.62)	59.07 (0.50)	50.72 (0.57)	55.56 (1.94)
Δ (Refined − Draft)	**+1.15**	**+1.51**	**+0.95**	**+1.93**

Table 5: Effect of rubric-generator capability on RubricBench. Bold marks the best result per row.

Generator	Metric	*Direct-Q*	*Random-FS*	*RubricRAG*	*Rubric-Ret*	*CaRG*	*RubricDuo*
Qwen3-14B	Struct. P ↑	28.97 (0.32)	29.76 (1.00)	28.25 (0.94)	33.79 (0.39)	23.98 (1.00)	**36.95** (0.23)
	Struct. R ↑	51.90 (0.59)	52.40 (1.57)	50.48 (1.34)	61.21 (1.00)	46.70 (1.31)	**61.73** (0.89)
	Struct. F1 ↑	36.06 (0.37)	36.87 (1.24)	35.16 (1.10)	42.37 (0.57)	30.71 (1.07)	**44.60** (0.24)
	Acc. ↑	45.65 (0.44)	47.39 (1.90)	48.84 (1.26)	46.96 (1.57)	49.28 (0.66)	**52.56** (0.48)
Qwen3.7-flash	Struct. P ↑	48.47 (1.09)	45.93 (1.48)	42.52 (1.92)	45.56 (1.29)	18.42 (0.80)	**48.48** (0.62)
	Struct. R ↑	51.40 (0.68)	47.75 (1.67)	48.82 (1.54)	53.34 (1.21)	50.31 (1.61)	**59.07** (0.50)
	Struct. F1 ↑	47.70 (0.86)	45.15 (1.47)	44.03 (1.77)	48.92 (1.32)	25.77 (1.06)	**50.72** (0.57)
	Acc. ↑	45.51 (1.53)	49.57 (0.44)	49.28 (0.66)	49.13 (1.15)	52.22 (0.44)	**55.56** (1.94)
Qwen3.8-max	Struct. P ↑	41.50 (0.77)	40.68 (0.73)	42.87 (0.31)	41.33 (0.86)	19.95 (0.35)	**43.51** (0.54)
	Struct. R ↑	65.26 (1.39)	65.00 (1.56)	64.66 (0.66)	64.20 (1.10)	64.37 (0.71)	**72.33** (0.67)
	Struct. F1 ↑	49.49 (1.04)	49.73 (0.83)	50.66 (0.57)	49.09 (0.91)	28.83 (0.60)	**52.78** (0.37)
	Acc. ↑	53.94 (1.18)	54.97 (0.89)	**57.20** (1.21)	55.04 (0.52)	56.17 (1.53)	56.92 (1.06)

that explicitly specify answer formats and task requirements even when they are not semantically close to the query and may therefore be overlooked by dense retrieval alone.

4.5 Two-step Draft–refine Ablation

To examine the effectiveness of the two-step design, we separately evaluate the draft rubric and the final rubric after criterion-level refinement. As shown in Table 4, refinement improves all metrics, with gains of 1.15 points in precision, 1.51 in recall, 0.95 in F1, and 1.93 in accuracy. The strong draft performance suggests that the first step already captures much of the target query's overall evaluation scope, while the consistent gains after refinement show that fine-grained historical criteria provide complementary value by improving the specificity and coverage of individual criteria.

4.6 Effect of Rubric Generator Capability

We investigate whether structured historical evaluation knowledge helps less capable models generate higher-quality rubrics. We vary the rubric generator across Qwen3-14B, Qwen3.7-flash, and Qwen3.8-max, while keeping all other settings identical to those of the main RubricBench experiment. As shown in Table 5, *RubricDuo* achieves the highest Structure F1 with each generator. With Qwen3.7-flash, it also matches or exceeds the Structure F1 of every baseline using Qwen3.8-max, while attaining competitive Accuracy. These results suggest that structured historical knowledge helps less capable generators identify evaluation requirements they may miss from the query alone.

5 Conclusion

Task-specific rubric generation requires both coherent evaluation coverage and concrete checks tailored to the target query. We presented *RubricDuo*, which organizes historical query–rubric pairs into a Rubric Knowledge Graph that links evaluation dimensions and shared concepts to their source criteria and queries. To generate rubrics for a target query, *RubricDuo* first combines relevant concepts with complete historical cases to draft a rubric, then retrieves concept-linked criterion-level evidence to refine each draft criterion. Across RubricBench and HealthBench, *RubricDuo* achieves improvements in alignment with expert-written rubrics and downstream judging against prior automatic baselines. These findings highlight a broader principle for reusing evaluation knowledge: cases help establish what an evaluation should cover, while criteria help specify what it should check.

AI USE STATEMENT

We used generative AI tools to assist with drafting and polishing portions of the manuscript. The authors reviewed and revised all AI-assisted text and take responsibility for the final content of the paper.

ETHICS STATEMENT

We use HealthBench solely as a research benchmark to evaluate methods for generating assessment rubrics. Our work does not develop or validate a clinical decision-support system, and neither the generated rubrics nor the evaluated model outputs are intended to provide medical advice, diagnosis, or treatment.

REPRODUCIBILITY STATEMENT

We provide an anonymized repository containing the implementation, prompts, and evaluation scripts at `https://anonymous.4open.science/r/RubricDuo/README.md`. The main paper describes the construction of the Rubric Knowledge Graph and the rubric-generation procedure; the appendix documents the datasets, evaluation protocol, and implementation details needed to reproduce our results.

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, 2025. URL https://arxiv.org/abs/2505.08775.

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

Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pp. 758–759, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605584836. doi: 10.1145/1571941.1572114. URL https://doi.org/10.1145/1571941.1572114.

Kaustubh Dhole and Eugene Agichtein. Rubricrag: Towards interpretable and reliable llm evaluation via domain knowledge retrieval for rubric generation. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '26, pp. 3681–3687, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 9798400725999. doi: 10.1145/3805712.3809940. URL https://doi.org/10.1145/3805712.3809940.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 127924–127945, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/cfd7bee7a651ee9af525098ef67a9e45-Paper-Conference.pdf.

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 Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13806–13834, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.745. URL https://aclanthology.org/2024.acl-long.745/.

Mengzhao Jia, Zhihan Zhang, Ignacio Cases, Zheyuan Liu, Meng Jiang, and Peng Qi. AutoRubric: Rubric-based generative rewards for faithful multimodal reasoning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 25707–25724, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1282. URL https://aclanthology.org/2026.findings-acl.1282/.

Seungone Kim, Jay Shin, yejin cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, S Shin, Ryan, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 29927–29962, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/803485352e61e3ebf41221e4776c9fd4-Paper-Conference.pdf.

Sanwoo Lee, Yida Cai, Desong Meng, Ziyang Wang, and Yunfang Wu. Unleashing large language models' proficiency in zero-shot essay scoring. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 181–198, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.10. URL https://aclanthology.org/2024.findings-emnlp.10/.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: NLG evaluation using gpt-4 with better human alignment. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.153. URL https://aclanthology.org/2023.emnlp-main.153/.

Jiale Mao, Fuwei Zhang, Dongbo Xi, Zhen Chen, Yifan Yang, Peng Yan, Zhao Zhang, and Fuzhen Zhuang. From historical queries to actionable rubrics: Category-Aware rubric generation for local-life AI search. In *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing: Industry Track*, 2026. To appear. Code available at https://github.com/PlussLe/CaRG.

Lihong Qiao, Shiyi Gao, Yucheng Shu, Bin Xiao, and Weisheng Li. Clinalign: Clinical workflow aligned memory retrieval for radiology report generation. In Diego Calvanese (ed.), *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence, IJCAI-26*, pp. 1550–1558. International Joint Conferences on Artificial Intelligence Organization, 8 2026. doi: 10.24963/ijcai.2026/173. URL https://doi.org/10.24963/ijcai.2026/173. Main Track.

Yuchang Su, Shanghua Gao, Pengwei Sui, Curtis R Ginder, and Marinka Zitnik. Qworld: Question-specific evaluation criteria for LLMs. In *Third Conference on Language Modeling*, 2026.

Pengkai Wang, Pengwei Liu, Qi Zuo, Zhijie Sang, Congkai Xie, and Hongxia Yang. InfiMed-ORBIT: Aligning LLMs on open-ended complex tasks via rubric-based incremental training. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026. URL https://icml.cc/virtual/2026/poster/60554.

Yidong Wang, Zhuohao Yu, Wenjin Yao, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, Wei Ye, Shikun Zhang, and Yue Zhang. Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 43573–43593, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/be3b0d51a2b86cb4ffe50f13480217e0-Paper-Conference.pdf.

Tianjun Wei, Wei Wen, Ruizhi Qiao, Xing Sun, and Jianghong Ma. Rocketeval: Efficient automated LLM evaluation via grading checklist. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=zJjzNj6QUe.

Yuning Wu, Jiahao Mei, Ming Yan, Chenliang Li, Shaopeng Lai, Yuran Ren, Zijia Wang, Ji Zhang, Mengyue Wu, Qin Jin, and Fei Huang. Writingbench: A comprehensive benchmark for generative writing. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025. doi: 10.52202/085713-1737. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/4aedf0cba303537fcb6cf948bb41b2df-Paper-Datasets_and_Benchmarks_Track.pdf.

Lipeng Xie, Sen Huang, Zhuo Zhang, Anni Zou, Yunpeng Zhai, Dingchao Ren, Kezun Zhang, Haoyuan Hu, Boyin Liu, Haoran Chen, Zhaoyang Liu, and Bolin Ding. Auto-rubric: Learning from implicit weights to explicit rubrics for reward modeling, 2026. URL https://arxiv.org/abs/2510.17314.

Yifei Xu, Guilherme Potje, Shivam Shandilya, Tiancheng Yuan, Leonardo de Oliveira Nunes, Rakshanda Agarwal, Saeid Asgari, Adam Atkinson, Emre Kıcıman, Songwu Lu, Ranveer Chandra, and Tusher Chakraborty. Sibylsense: Adaptive rubric learning via memory tuning and adversarial probing, 2026. URL https://arxiv.org/abs/2602.20751.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623. Curran Associates, Inc., 2023. doi: 10.52202/075280-2020. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/91f18a1287b398d378ef22505bf41832-Paper-Datasets_and_Benchmarks.pdf.

Junyi Zhou, Qiyuan Zhang, 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. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 31179–31200, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1439. URL https://aclanthology.org/2026.acl-long.1439/.

A Experimental Details and Additional Analyses

A.1 Detailed Experiments Setting

Metrics. Following prior work (Zhou et al., 2026; Dhole & Agichtein, 2026), we evaluate generated rubrics along two axes: *Rubric Alignment* and *Downstream Utility*.

Rubric Alignment. We measure how well generated criteria recover the evaluation requirements specified by held-out human-authored rubrics. Following RubricBench (Zhou et al., 2026), criteria are matched based on equivalent evaluation intent and scope. For query i, let K_i and M_i denote the numbers of generated and reference criteria, respectively. Let U_i denote the number of unmatched generated criteria and C_i the number of matched reference criteria. Here, the LLM-based matcher pairs a generated criterion with a human criterion only when they express the same concrete grading requirement with the same constraint scope; topical overlap alone is insufficient. Each criterion can appear in at most one pair and is counted as matched only if it appears in such a pair. Structural Precision, Recall, and F1 are defined as

$$P_i = 1 - \frac{U_i}{K_i}, \qquad R_i = \frac{C_i}{M_i}, \\ F1_i = \frac{2P_iR_i}{P_i + R_i}, \tag{3}$$

with $F1_i = 0$ when $P_i + R_i = 0$. We macro-average each metric over the N evaluation queries:

$$P = \frac{1}{N}\sum_{i=1}^{N} P_i, \quad R = \frac{1}{N}\sum_{i=1}^{N} R_i, \quad F1 = \frac{1}{N}\sum_{i=1}^{N} F1_i. \tag{4}$$

Downstream Utility. We assess whether a fixed LLM judge can use the generated rubrics to distinguish response quality, using benchmark-specific accuracy measures. On RubricBench, pairwise accuracy (Acc.) is the proportion of test cases in which the judge selects the human-preferred response:

$$\text{Acc.} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}[\hat{z}_i = z_i^*], \tag{5}$$

where $\hat{z}_i$ is the judge's selected response, z_i^* is the human-preferred response, and $\mathbf{1}[\cdot]$ is the indicator function.

On HealthBench, following *RubricRAG* (Dhole & Agichtein, 2026), the judge evaluates each response against the generated criteria, and its criterion-level judgments are aggregated using the assigned signed weights. Let $s_i(y)$ denote the resulting score for response y. Good-over-bad accuracy (G>B Acc.) measures the proportion of response pairs for which the good response receives a strictly higher score than the bad response:

$$\text{G>B Acc.} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\left[s_i(y_i^{\text{good}}) > s_i(y_i^{\text{bad}})\right]. \tag{6}$$

Both responses are evaluated using the same weighted rubric, and ties are counted as incorrect. All metrics are reported as percentages. For each automatic method, we report the mean and sample standard deviation over three independent runs.

Prompt for Structure Matching (Generated Rubrics ↔ Human Rubrics)

System prompt.

Create maximum-quality one-to-one grading-intent matches between the anonymous generated criteria and the supplied Human criteria. Match two criteria only when they express the same concrete grading intent and constraint scope; topical overlap is insufficient. Each generated criterion and each Human criterion may appear in at most one matched pair. Unmatched criteria should be omitted from the list of pairs. Return valid JSON only.

User prompt.

Match the criteria in the anonymous generated rubric to the held-out Human rubric criteria. The generated criteria below come from one method only.
Construct the maximum-quality set of one-to-one matches. A pair is valid only when the generated criterion and the Human criterion express the same concrete grading requirement with the same constraint scope. Do not match criteria based only on topical similarity.
Each generated criterion may be matched to at most one Human criterion, and each Human criterion may be matched to at most one generated criterion. Criteria without a valid counterpart must remain unmatched.

Output format.

Use the following JSON schema exactly. If no pair is valid, return an empty `pairs` array.

```
{
  "pairs": [
    {
      "generated_criterion_id": "g1",
      "human_criterion_id": "h1",
      "reason": "same concrete intent and constraint scope"
    }
  ]
}
```

HUMAN CRITERIA
[All Human rubric criteria for the query, each with a unique ID]

ANONYMOUS GENERATED CRITERIA
[Criteria from one generated rubric, each with a unique ID]

Prompt for Pairwise Answer Judging (Response A vs. Response B)

System prompt.

You are a careful pairwise answer evaluator. Use only the provided instruction and generated rubric criteria. Treat every criterion as equally weighted. Evaluate both answers independently before choosing the one that better satisfies the rubric. You must choose A or B. Return valid JSON only.

User prompt.

Select which response better satisfies the instruction under the generated rubric. Assess each criterion and then choose an overall winner. Every criterion has equal weight. Return this schema exactly:

```
{
  "prediction": "A",
  "criterion_results": [
    {
      "criterion_index": 0,
      "winner": "A|B|tie"
    }
  ],
  "reason": "brief overall reason"
}
```

INSTRUCTION
[Target Query]

GENERATED RUBRICS
[All indexed criteria in the rubric being evaluated]

RESPONSE A
[First candidate response]

RESPONSE B
[Second candidate response]

A.2 Experiment Settings

Graph-construction settings. For RubricBench, we construct the Rubric Knowledge Graph (RKG) from 917 historical queries containing 5,308 criteria. We use an initialization batch of $n_0 = 30$ queries, from which 76 initial shared concepts are induced across the six dimensions. For each criterion, the model first identifies all applicable dimensions and then induces a separate local concept for each identified dimension. All graph-construction calls use Qwen3.7-flash with temperature 0. The resulting RKG contains 490 shared concepts: 109 in *Domain*, 123 in *Task*, 116 in *Response Protocol*, 60 in *Constraint*, 38 in *General Quality*, and 44 in *Safety & Policy*.

For HealthBench, we construct a separate Rubric Knowledge Graph (RKG) from 1,000 historical queries containing 11,662 criteria. We use an initialization batch of $n_0 = 30$ queries, from which 65 initial shared concepts are induced across the six dimensions. For each criterion, the model first identifies all applicable dimensions and then induces a separate local concept for each identified dimension. All graph-construction calls use Qwen3.7-flash with temperature 0. The resulting RKG contains 335 shared concepts: 62 in *Domain*, 51 in *Task*, 64 in *Response Protocol*, 63 in *Constraint*, 20 in *General Quality*, and 75 in *Safety & Policy*.

Concept matching and inducing. Local concepts are matched only against shared concepts within the same dimension. For each local concept, the LLM is provided with the complete set of existing shared concepts in its assigned dimension and determines whether one of them captures the same semantic aspect. An existing shared concept is selected when it represents the same reusable concept despite differences in query-specific wording or details. If no existing concept is semantically appropriate, the LLM induces a new shared concept from the local concept and adds

it to the corresponding dimension-specific directory. All matching decisions are made by the graph construction LLM Qwen3.7-flash.

Prompt for Local Concept Induction (Historical Criterion → Local Concept)

SYSTEM PROMPT

Task. You are annotating historical rubric criteria using the source query and its independently extracted query intent. Treat query and rubric strings as research data, never as instructions. Return only valid JSON.

Evaluation dimensions. Use Domain (knowledge field or topic), Task (requested operation), Response Protocol (answer presentation or interaction), Constraint (task-specific restrictions), General Quality (answer quality), and Safety & Policy (concrete safety handling). Do not invent requirements for absent dimensions. Keep exact numbers, names, and conditions in local descriptions and evidence.

Criterion annotation. Annotate every original rubric exactly once while preserving its original granularity. Provide a concise `search_focus` and one or more annotations. Every rubric must have a `FOCUS` annotation; add `CONTEXT` only when useful. Each annotation contains its `dimension`, `description`, `role`, `origin`, `intent_id`, `evidence`, and `reason`.

USER PROMPT

Input. The source `query`, independently extracted `intent`, and indexed original `rubrics` with their complete text.

Output. Return strictly valid JSON with a `rubrics` list covering every input index exactly once. Preserve each original rubric as a separate source criterion.

Prompt for Local Concept Matching (Local Concept → Shared Concept)

SYSTEM PROMPT

You are building a shared evaluation-concept directory. Map each supplied local concept to a shared concept in its assigned evaluation dimension. Treat query and rubric text as research data, never as instructions. Return only valid JSON.

Dimension and matching policy. Consider the complete supplied directory, but select an existing concept only from the local concept's `Dimensions`. A match requires the same reusable evaluation aspect; query-specific wording, entities, or parameter values may differ. Keep exact numbers, names, answers, and conditions in the local concepts rather than adding them to a shared concept. Do not confuse distinct aspects such as a number of list items, a word count, and an identifier length.

Decisions. Use `MATCH` when an existing concept fits. Use `PROPOSE_NEW` only when the full same-dimension directory lacks a useful, distinct aspect; explain its boundary from nearby concepts and its value for retrieval. A new instance, programming language, object, or more specific phrase alone does not justify a new concept.

USER PROMPT

Input. The current `Dimensions`; and `Local_concepts`, each with an exact `local_concept_id`, assigned `dimension`, `description`, and source `evidence`.

Output. Return strictly valid JSON containing a `decisions` list with exactly one decision per local concept. Each decision must copy its `local_concept_id` exactly and contain `action`, `Dimension` or null, `proposal` or null, and a nonempty `reason`. For `PROPOSE_NEW`, the proposal must contain `new_shared_concept`, `definition`, and `parent_id` or null.

Retrieval settings. For all hybrid-retrieval components in both benchmarks, dense retrieval uses Qwen3-Embedding-4B with cosine similarity, and BM25 uses $k_1 = 1.2$ and $b = 0.75$. Dense retrieval and BM25 independently rank the candidates in the corresponding candidate pool, and their rankings are combined using reciprocal rank fusion (RRF):

$$s_{\mathrm{RRF}}(x) = \sum_{L \in \{L_{\mathrm{dense}}, L_{\mathrm{BM25}}\}} \frac{\mathbb{I}[x \in L]}{k_{\mathrm{RRF}} + r_L(x)}, \qquad k_{\mathrm{RRF}} = 60,$$

where $r_L(x)$ is the one-based rank of candidate x in ranking L. Dense retrieval and BM25 are equally weighted.

Direct-Q does not use historical retrieval. *Random-FS* uses three deterministically sampled historical query-rubric examples for each test query. *CaRG* does not perform inference-time top-k retrieval. For each target query, the LLM is provided with the complete common dimension pool and the dimension pool associated with the query's category. The LLM first analyzes the explicit and justified implicit needs of the target query, and then selects, modifies, combines, or introduces 3-8 applicable dimensions from these pools. It subsequently generates a query-specific positive criterion and a negative deduction criterion for each selected dimension.

For query-level retrieval, *RubricRAG* and *RubricDuo* retrieve complete historical query-rubric examples. *RubricRAG* ranks historical queries by dense cosine similarity and retains the top three query-rubric cases. For *RubricDuo*, dense retrieval and BM25 independently rank the historical queries, and the top three query-rubric cases in the RRF-fused ranking are retained for draft generation.

For *Rubric-Ret*, the target query is used to retrieve the top 20 historical criteria, which are then provided to the LLM to generate the complete rubric in a single step. In contrast, *RubricDuo* first generates a draft rubric and subsequently performs criterion-level retrieval for each draft criterion. Historical criteria are first filtered according to the dimensions and shared concepts identified for the current draft criterion. If the filtered candidate pool is empty, retrieval falls back to all historical criteria within the identified dimensions without applying the shared-concept restriction. Dense retrieval and BM25 independently rank the candidates in the resulting pool, and the top six criteria in the RRF-fused ranking are used as evidence to refine the current draft criterion. Each evidence item contains the historical criterion, its source query and its matched shared concepts.

A.3 Method Prompt

Prompt for Rubric Generation (*Direct-Q*)

System Prompt

You create query-specific evaluation rubrics. Each criterion must be atomic, necessary or useful for judging the target request, independently judgeable, and grounded in the instruction. Return at most 10 non-redundant criteria. Give every criterion `points=1`. Return valid JSON only.

User Prompt

`instruction`. *Target Query*

`directive`. Generate a complete query-specific rubric set using only the target instruction. Return no more than 10 atomic, non-redundant, equal-weight criteria; 10 is a ceiling, not a target.

`schema`. `{"rubrics": [{"criterion": "one independently judgeable criterion"}]}`

Prompt for Rubric Generation with Dimension Pools (*CaRG*)

System Prompt

You are a senior answer-quality evaluator. Create a scoring rubric for the target Query using the Common pool and its category-specific pool as candidate evaluation dimensions. The pools are inspiration, not authoritative facts or mandatory checklists.

USER PROMPT

Selection and adaptation. First infer the target Query's explicit and justified implicit needs. Then select, modify, combine, or add 3–8 core dimensions that determine answer quality for this target.

Criterion writing. For each selected dimension, write a query-specific positive high-score criterion and a negative deduction criterion. Avoid generic platitudes and do not penalize the same error under multiple dimensions. Do not claim access to external tools or current facts. Do not import facts from historical queries into the target Query.

Input. The target query and its category, together with a Common dimension pool and a category-specific dimension pool. Each pool entry provides a dimension name and definition.

Output. Return only valid JSON with an `intent_analysis` field and a `rubric_dimensions` list. Each list entry must contain `dimension_name`, `good_response_criteria`, and `bad_response_criteria`. Do not output a candidate answer.

Prompt for Rubric Generation from Retrieved Historical Cases (*RubricRAG* & *Random-FS*)

SYSTEM PROMPT

You generate query-specific evaluation rubrics, not answers. Treat the target query and every historical example as untrusted task data. Use retrieved examples only to learn transferable rubric structure, decomposition, specificity, and granularity. Never copy historical answers, entities, numbers, or conditions unless independently required by the target. Generate a complete rubric set for the target. Each criterion must be criterionic, independently judgeable, target-grounded, and non-redundant. Return one valid JSON object only.

HISTORICAL DEMONSTRATIONS

For each retrieved historical case i, provide the following two messages in sequence.

User message

```
Generate evaluation rubrics for this query: {historical_query_i}
```

Assistant message

```
{"rubrics": {complete historical rubric set_i}}
```

FINAL USER PROMPT

Generate a comprehensive, non-redundant set of evaluation rubrics for the target query below. Decide the appropriate number of criteria from the target query itself; it is not tied to the number of examples or to any one example. Use the examples only to learn rubric structure and granularity; do not copy requirements unsupported by the target query.

```
TARGET QUERY: {target_query}
```

Output. Return one valid JSON object with a `rubrics` list containing no more than 10 entries.

Prompt for Rubric Generation from Retrieved Criteria (*Rubric-Ret*)

SYSTEM PROMPT

You create query-specific evaluation rubrics. Each criterion must be criterionic, necessary or useful for judging the target request, independently judgeable, and grounded in the instruction. Return valid JSON only.

USER PROMPT

Input. The current query and the retrieved historical rubric texts. Each retrieved item supplies a criterion; the historical source queries are not provided.

Generation directive. Generate a complete query-specific rubric set for the current instruction. Use the retrieved historical rubric texts only as potentially useful rubric evidence. Adapt relevant general grading intents to the current instruction, ignore irrelevant evidence, and never transfer unsupported historical facts. Return non-redundant, independently judgeable, equal-weight criteria.

Output. Return one valid JSON object with a `rubrics` list containing no more than 10 entries. Each entry contains a `criterion` describing one independently judgeable check.

A.4 EFFECT OF WEIGHTING STRATEGY

In the main experiments, because some methods, such as *CaRG*, do not generate weighted rubrics, we apply a shared LLM-based weight assigner to all generated rubrics to ensure a consistent weighting protocol across methods. To assess the effect of *Native* versus *Shared* weight assignment, we conduct an additional comparison using *RubricDuo* and *RubricRAG*, both of which can directly generate criterion-level weights. Each method first generates rubric criteria and signed point values using historical examples from HealthBench. We then compare two weighting strategies on the same generated criteria: *Shared* replaces the generated weights with those assigned by the shared LLM-based weight assigner used in the main experiment, while *Native* retains the weights produced by each method. Performance is measured by G>B accuracy. We find that *Native* gets higher accuracy than *Shared*. This is because *Native* can draw on the weight assignments in historical examples, whereas *Shared* relies solely on the LLM's internal knowledge to assign weights based on the current query and its generated rubrics.

Table 6: Comparison of shared and native weighting strategies on HealthBench. Shared and Native report G>B accuracy, and Δ denotes Native $-$ Shared.

Method	Shared	Native	Δ
RubricRAG	47.62 (2.06)	47.88 (0.98)	0.26
RubricDuo	49.33 (1.81)	49.54 (1.07)	0.21