SURE: Evolving Pairwise Reward Models with Rubrics from Uncertain Comparisons

Anonymous authors
Paper under double-blind review

Abstract

Pairwise reward models (RMs) are widely used as scalable proxies for human preferences in language model alignment, representing the relative quality of two responses through their reward difference. However, their judgments can become unreliable under distribution shift, particularly for comparisons near the model's decision boundary or requiring criteria and domain knowledge insufficiently covered during training. We propose `SURE`, a sample-efficient framework for evolving pairwise RMs with rubrics learned from uncertain comparisons. `SURE` identifies informative comparisons from an unlabeled target set using two complementary signals calibrated against in-distribution anchors: reward-margin uncertainty and representation-space deviation. It queries a stronger LLM only for the selected pairs and uses the resulting judgments to re-train the RM, while deriving hierarchical rubrics containing general principles and domain-specific criteria from the accompanying rationales. The rubrics are further refined through Group Relative Policy Optimization (GRPO)-inspired textual optimization, which converts group-relative agreement with the stronger LLM into natural-language feedback rather than parameter gradients. At inference time, the evolved rubrics provide explicit criteria for preference judgment, complementing the adapted RM. Experiments show that, under matched selection ratios or external-query budgets, `SURE` identifies more informative uncertain cases, improves pairwise preference accuracy under distribution shift, and delivers reasonable gains on downstream tasks. Our code is at `https://anonymous.4open.science/status/SURE_ICLR` and associated models will be released publicly.

1 Introduction

Pairwise reward models play a central role in preference alignment for modern large language models (LLMs) (Gao et al., 2023). In reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022; Lambert et al., 2025) and related alignment pipelines (Christiano et al., 2017; Stiennon et al., 2020), they compare candidate responses to the same prompt and provide scalable preference signals in place of repeated human annotation. Compared with pointwise reward models that score responses independently, pairwise RMs better match how human preferences are collected, avoid calibrating absolute scores across heterogeneous prompts, and focus directly on relative response quality (Zheng et al., 2023; Liusie et al., 2024). Nevertheless, they must still produce a preference when a comparison is ambiguous, distributionally shifted, or governed by criteria insufficiently represented in their training data, which leads to unreliable preference prediction results.

This limitation is particularly consequential in open-ended preference evaluation (Liu et al., 2023; Lu et al., 2024; Rafailov et al., 2023), where response quality may depend simultaneously on correctness, usefulness, reasoning quality, safety, instruction following, and domain-specific conventions. When candidate responses are semantically close or require unfamiliar knowledge, an RM may produce an apparently decisive reward difference despite an unreliable judgment. The challenge is therefore to **identify where its preference boundary is unreliable** and turn a limited number of such cases **into supervision** that improves performance across the target distribution (Ye et al., 2024). A natural source of such supervision is a stronger LLM, which can provide more reliable preference judgments and rationales but is substantially more expensive to query. This motivates selectively querying it only for the most uncertain comparisons.

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.

Reward-model uncertainty provides a natural basis for this selection. A small reward margin indicates proximity to the model's decision boundary, whereas representation-space deviation from in-distribution anchors indicates potential distribution shift[1]. We calibrate both signals against anchors sampled from the original RM training distribution and combine them to rank an unlabeled target set. This allows us to focus the limited query budget for the stronger LLM (Wang et al., 2024) on comparisons that are most informative for the current RM.

Based on this observation, we propose SURE (**S**electing **U**ncertain Comparisons for **R**eward-model **E**volution). SURE queries a stronger LLM only for the selected comparisons. The stronger LLM's preference judgments provide targeted supervision for retraining the pairwise RM, while the accompanying rationales (Ke et al., 2024) are used to derive a set of hierarchical rules—referred to as rubrics—that guide the reward model's judgments. A general layer captures domain-invariant principles such as correctness and usefulness, while domain-specific layers capture requirements such as mathematical rigor, code executability, and appropriate refusal behavior.

The obtained rubrics are further refined through GRPO-inspired textual optimization. Rubric-conditioned judgments are compared with judgments from the stronger LLM, and agreement scores are normalized within each group to obtain relative-advantage-like signals. These signals organize natural-language feedback about criteria that should be retained, revised, or introduced; they are not back-propagated into model parameters. In this way, the samples selected by the uncertainty criteria proposed in SURE serve a dual purpose: they are used both for **continued training** of the original RM and for **summarizing and evolving the rubrics** that can be used for preference judgment at inference time. Based on SURE, we evaluate the results on preference benchmarks and downstream alignment tasks, showing that uncertainty-guided selection uses matched data and query budgets more effectively while improving preference accuracy under distribution shift.

Overall, our contributions are: (i) We introduce SURE, a sample-efficient framework that uses RM uncertainty to select informative target comparisons and jointly improve pairwise RMs and their preference rubrics from feedback provided by a stronger LLM. (ii) We combine anchor-calibrated reward-margin uncertainty and representation-space deviation to capture decision-boundary ambiguity and distribution shift. (iii) We use judgments from the stronger LLM for targeted RM re-training and derive hierarchical rubrics from their rationales, refining the rubrics through GRPO-inspired group-relative textual feedback. (iv) We show that SURE improves pairwise preference accuracy under matched selection ratios or query budgets and yields reasonable gains on downstream tasks.

2 Related Work

2.1 Reward Models

Reward modeling originates from the Bradley–Terry pairwise comparison model (Bradley & Terry, 1952); Christiano et al. (2017) trained reward predictors from human pairwise preferences over trajectory segments for reinforcement learning, and subsequent work adapted this idea to language modeling (Stiennon et al., 2020; Bakker et al., 2022; Bai et al., 2022; Jiang et al., 2023; Wu et al., 2023). Existing reward models can be broadly divided into point-wise and pairwise formulations. Point-wise reward models assign an absolute scalar score to each response (Matrenok et al., 2026), which is convenient for ranking and policy optimization, but absolute-score uncertainty is not well-defined[2]. Pairwise reward models instead compare two responses to the same prompt and induce preferences through their reward difference (Zhang et al., 2025), better matching the format of human preference annotation. This pairwise formulation also provides a natural setting for comparison (Chen & Tan, 2026). However, prior work mostly treats reward-model uncertainty as a confidence or routing signal (Xu et al., 2026). In contrast, SURE leverages uncertainty as an interface for both sample selection and rubric evolution, thereby improving model performance.

2.2 Rubric Evolution

Rubric-based evaluation aims to make the criteria behind open-ended judgments explicit, improving both controllability and interpretability of model evaluation (Zhong et al., 2022). Prior work has used

[1] Detailed experimental evidence is provided in Appendix C.3.

[2] More explanations can be found in Appendix A.

natural-language criteria, form-filling protocols, and customized rubrics to guide LLM evaluators, showing that explicit standards can make judgments more structured and less dependent on implicit evaluator preferences (Liu et al., 2023; Kim et al., 2024a;b). A related line of work treats prompts, instructions, or textual feedback as optimizable objects, using model-generated critiques or textual gradients to iteratively refine natural-language components (Zhou et al., 2023; Pryzant et al., 2023; Yang et al., 2024; Yuksekgonul et al., 2025). SURE follows this direction but shifts the focus from optimizing generic prompts to evolving evaluation rubrics: reward-model uncertainty first identifies comparisons where evaluative criteria are missing or unstable, and stronger-LLM rationales are then distilled into hierarchical rubrics and refined through group-relative textual feedback.

3 Preliminaries and Problem Definitions

We consider the problem of preference evaluation of reward models for large language model outputs. Given an input prompt $x \in \mathcal{X}$, a language model may produce multiple candidate responses $y \in \mathcal{Y}$. A preference evaluator compares these responses and determines which one better satisfies the user's intent. In this work, we focus on pairwise preference modeling, where the evaluator receives a prompt and two candidate responses and predicts a binary preference.

3.1 Pairwise Preference Modeling

Let $\mathcal{D} = \{(x_i, y_i^+, y_i^-)\}_{i=1}^N$ denote a preference dataset, where y_i^+ is preferred over y_i^- for prompt x_i. A point-wise reward model assigns a scalar score to each prompt-response pair $r_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. The pairwise preference between two responses is induced by their reward difference:

$$\Delta_\theta(x, y_a, y_b) = r_\theta(x, y_a) - r_\theta(x, y_b), \tag{1}$$

where the model predicts y_a to be preferred over y_b when $\Delta_\theta(x, y_a, y_b) > 0$, and predicts y_b otherwise. And the standard training objective for pairwise reward modeling is the Bradley–Terry loss (Bradley & Terry, 1952):

$$\mathcal{L}_{\mathrm{BT}}(\theta) = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}} \left[\log \sigma \left(r_\theta(x, y^+) - r_\theta(x, y^-)\right)\right], \tag{2}$$

where $\sigma(\cdot)$ denotes the sigmoid function. This objective encourages the reward assigned to the preferred response to be larger than that assigned to the rejected response.

3.2 Budgeted Preference Learning under OOD Shift

We study pairwise reward modeling under target-distribution shift. Let $\mathcal{D}_{\mathrm{ID}}$ denote the in-distribution data used to train or calibrate a reward model r_θ, and let $\mathcal{D}_{\mathrm{test}} = \{(x_i, y_{i,a}, y_{i,b})\}_{i=1}^M$ be an unlabeled target set of pairwise comparisons. The target set may contain shifted domains, harder reasoning cases, or response patterns not sufficiently covered by $\mathcal{D}_{\mathrm{ID}}$.

Our goal is to improve preference judgment on the full target set while querying a stronger LLM for only a small subset of examples. Under an annotation budget $B \ll M$, we select the most informative comparisons according to an uncertainty score $s(x, y_a, y_b)$, forming $\mathcal{S} = \mathrm{TopK}_{(x,y_a,y_b)\in\mathcal{D}_{\mathrm{test}}} s(x, y_a, y_b)$, where $|\mathcal{S}| \le B$.

The stronger LLM provides judgments and rationales only for $\mathcal{S}$. We use this feedback to adapt the reward model and construct a natural-language rubric $q = \mathrm{RubricGen}(\mathcal{S})$. Denoting the adapted reward model by $r_{\theta'}$, our objective is $\max_{\mathcal{S},q,\theta'} \mathrm{Acc}_{\mathcal{D}_{\mathrm{test}}}(r_{\theta'}, q)$, subject to $|\mathcal{S}| \le B$.

4 SURE Framework

We propose SURE (as shown in Figure 1) that uses reward-model uncertainty to (i) select informative comparisons, (ii) induce explicit rubrics, and (iii) re-train the preference model. It assumes a trained reward model and a stronger LLM, but no ground-truth labels—a practical setting where labels are unavailable or expensive and the stronger LLM can be queried selectively. SURE selects uncertain comparisons with the reward model, queries the stronger LLM for preference judgments and rationales on these comparisons, converts the rationales into general and domain-specific rubrics, and re-trains the preference model with the judgments and rubrics. These rubrics serve as explicit, reusable evaluation instructions for downstream preference judgment.

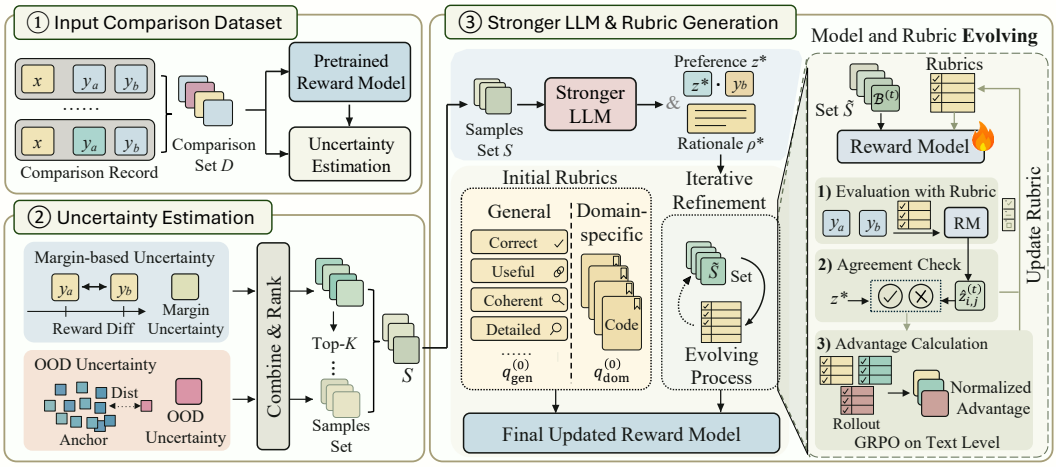


Figure 1: Overview of SURE. The framework selects informative comparisons using margin and OOD uncertainty, then leverages stronger-LLM feedback to re-train the reward model and evolve hierarchical rubrics.

4.1 Uncertainty-Based Sample Selection

4.1.1 Margin Uncertainty

The first stage aims to identify comparisons on which the reward model is likely to be unreliable. Given an input prompt x and two candidate responses y_a and y_b, the reward model assigns scalar scores $r_\theta(x, y_a)$ and $r_\theta(x, y_b)$. The induced preference is determined by the reward difference

$$\Delta_\theta(x, y_a, y_b) = r_\theta(x, y_a) - r_\theta(x, y_b). \tag{3}$$

A small absolute margin indicates that the comparison lies near the reward model's decision boundary. We therefore define a margin-based uncertainty score as $u_{\text{margin}}(x, y_a, y_b) = -|\Delta_\theta(x, y_a, y_b)|$, where higher values correspond to lower confidence.

4.1.2 Distributional Uncertainty

Margin uncertainty captures ambiguity in the output space, but it does not determine whether a comparison is distributionally different from the reward model's training distribution. We therefore construct an in-distribution anchor set $\mathcal{A} = \{a_i\}_{i=1}^N$ by sampling comparisons from the dataset used to train the original reward model. Let $\mathbf{h}_\theta(x, y_a, y_b) \in \mathbb{R}^d$ denote the last-token hidden representation of a comparison. We estimate the anchor mean as $\boldsymbol{\mu}_\mathcal{A} = \frac{1}{N}\sum_{i=1}^N \mathbf{h}_\theta(a_i)$ and its covariance as $\boldsymbol{\Sigma}_\mathcal{A} = \frac{1}{N-1}\sum_{i=1}^N (\mathbf{h}_\theta(a_i) - \boldsymbol{\mu}_\mathcal{A})(\mathbf{h}_\theta(a_i) - \boldsymbol{\mu}_\mathcal{A})^\top$.

We then compute the squared Mahalanobis distance between a target comparison and the anchor distribution:

$$d_\mathcal{A}(x, y_a, y_b) = (\mathbf{h}_\theta(x, y_a, y_b) - \boldsymbol{\mu}_\mathcal{A})^\top (\boldsymbol{\Sigma}_\mathcal{A} + \lambda \mathbf{I})^{-1} (\mathbf{h}_\theta(x, y_a, y_b) - \boldsymbol{\mu}_\mathcal{A}), \tag{4}$$

where $\lambda > 0$ is a ridge regularization coefficient and $\mathbf{I} \in \mathbb{R}^{d\times d}$ is the identity matrix.

Let $F_\mathcal{A}^{\text{maha}}$ denote the empirical CDF of the squared Mahalanobis distances of the anchor samples:

$$F_\mathcal{A}^{\text{maha}}(t) = \frac{1}{N}\sum_{i=1}^N \mathbb{I}\left[d_\mathcal{A}(a_i) \le t\right].$$

The anchor-calibrated distributional uncertainty of a target comparison will be $u_{\text{ood}}(x, y_a, y_b) = F_\mathcal{A}^{\text{maha}}\left(d_\mathcal{A}(x, y_a, y_b)\right)$. Therefore, $u_{\text{ood}} \in [0, 1]$ measures the percentile of a target comparison's representation-space deviation relative to samples from the original reward-model training distribution. A larger value indicates stronger distributional shift. To make the two uncertainty signals comparable, we convert their raw values into empirical percentiles relative to the same anchor

set $\mathcal{A}$. We denote the resulting anchor-calibrated scores by $u_{\text{margin}}(v), u_{\text{ood}}(v) \in [0,1]$, where $v = (x, y_a, y_b)$ and larger values indicate greater uncertainty. We combine the two scores as:

$$u(v) = \alpha u_{\text{ood}}(v) + (1-\alpha) u_{\text{margin}}(v), \tag{5}$$

where $\alpha \in [0,1]$ balances distributional deviation and decision-boundary ambiguity. We then rank all target comparisons by $u(v)$ and select the top B comparisons:

$$\mathcal{S} = \text{TopK}_{v \in \mathcal{D}_{\text{test}}}\big(u(v); B\big), \qquad |\mathcal{S}| = B, \tag{6}$$

where $\text{TopK}(\cdot; B)$ returns the B comparisons with the highest combined uncertainty scores.

Furthermore, for each selected comparison $(x_i, y_{i,a}, y_{i,b}) \in \mathcal{S}$, SURE queries a stronger LLM to obtain a preference judgment $z_i^\star \in \{1,2\}$ and a textual rationale $\rho_i^\star$ as $(z_i^\star, \rho_i^\star) = g(x_i, y_{i,a}, y_{i,b})$, where $z_i^\star = 1$ indicates a preference for $y_{i,a}$ and $z_i^\star = 2$ indicates a preference for $y_{i,b}$. This produces the annotated selected set:

$$\widetilde{\mathcal{S}} = \{(x_i, y_{i,a}, y_{i,b}, z_i^\star, \rho_i^\star)\}_{i=1}^{|\mathcal{S}|}. \tag{7}$$

Compared with $\mathcal{S}$, the annotated set $\widetilde{\mathcal{S}}$ additionally contains the stronger LLM's preference judgments and rationales, which are subsequently used for reward-model re-training and rubric construction, respectively.

4.2 Rubric-Guided Reward Model Evolution

SURE uses the stronger-LLM-annotated set $\widetilde{\mathcal{S}}$ to re-train the pairwise reward model under an initial hierarchical rubric. The stronger LLM's judgments provide preference supervision, while its rationales inform the rubric's general and domain-specific criteria. After re-training, we hold the reward model fixed and refine the rubric through group-relative textual feedback. Candidate revisions are accepted only when they improve domain-balanced agreement with the stronger LLM on the current evolution batch. We describe initialization and evolution below.

4.2.1 Hierarchical Rubric Initialization

We initialize the rubric by summarizing reusable preference criteria from the judgments and rationales in $\widetilde{\mathcal{S}}$. Let $\mathcal{C}$ denote the set of domains. The initial hierarchical rubric consists of a general component and domain-specific components as $q^{(0)} = \left(q_{\text{gen}}^{(0)}, \{q_{\text{dom},d}^{(0)}\}_{d \in \mathcal{C}}\right)$. The general component captures shared principles, including correctness, usefulness, coherence, appropriate detail, and safety. Domain-specific components capture requirements such as mathematical rigor, code executability, and appropriate refusal. For a comparison from domain d, the effective instruction is:

$$q_d^{(0)} = q_{\text{gen}}^{(0)} \oplus q_{\text{dom},d}^{(0)}, \tag{8}$$

where $\oplus$ denotes concatenation of the general and domain-specific criteria. This structure separates shared preference principles from domain-sensitive rules. Note that both rubric initialization and evolution **use the base model RM adopted**, rather than the stronger LLM.

4.2.2 Re-training with $\widetilde{\mathcal{S}}$

Starting from the original parameters θ, we re-train the pairwise RM on $\widetilde{\mathcal{S}}$ with the initial rubric held fixed. For comparison i in domain d_i, the model receives $q_{d_i}^{(0)}$ as evaluative context and directly predicts a preference logit $s_i(\vartheta) = r_\vartheta(x_i, y_{i,a}, y_{i,b}; q_{d_i}^{(0)})$, where positive values favor $y_{i,a}$. With stronger-LLM judgments encoded as $z_i^\star \in \{1,2\}$, the training objective is

$$\mathcal{L}_{\text{train}}(\vartheta) = -\frac{1}{|\widetilde{\mathcal{S}}|} \sum_{i=1}^{|\widetilde{\mathcal{S}}|} \log \sigma\big((3 - 2z_i^\star) s_i(\vartheta)\big), \tag{9}$$

where σ is the sigmoid function and ϑ denotes the trainable parameters initialized at θ, and the resulting parameters are denoted by θ'. In this stage, the initial rubric supplies explicit criteria, while the stronger LLM supplies preference supervision. Based on the obtained θ', the rubric refinement stage is described below.

4.2.3 Advantage-Guided Rubric Evolution

After re-training, SURE freezes the reward-model parameters at θ' and refines the rubric through a textual optimization procedure inspired by Group Relative Policy Optimization (GRPO). The rubric-generation model is likewise frozen, and only the natural-language criteria evolve (Madaan et al., 2023; Fernando et al., 2023).

At iteration t, the parent rubric is $q^{(t)} = (q^{(t)}_{\text{gen}}, \{q^{(t)}_{\text{dom},d}\}_{d\in\mathcal{C}})$. Specifically, for domain d, its effective instruction will be $q^{(t)}_d = q^{(t)}_{\text{gen}} \oplus q^{(t)}_{\text{dom},d}$. Conditioned on the parent rubric, the annotated comparisons, and textual feedback from preceding iterations, the generation model produces J candidates $\{q^{(t)}_j\}_{j=1}^{J}$, whose effective instructions are $q^{(t)}_{j,d} = q^{(t)}_{j,\text{gen}} \oplus q^{(t)}_{j,\text{dom},d}$.

We evaluate both the parent and all candidate rubrics on the same annotated batch $\mathcal{B}^{(t)} \subseteq \widetilde{\mathcal{S}}$. For comparison i in domain d_i, the pairwise RM yields an order-symmetrized score:

$$\Delta^{(t)}_{i,j} = \frac{1}{2}\left[r_{\theta'}(x_i, y_{i,a}, y_{i,b}; q^{(t)}_{j,d_i}) - r_{\theta'}(x_i, y_{i,b}, y_{i,a}; q^{(t)}_{j,d_i})\right]. \tag{10}$$

Given preference labels encoded as $z^\star_i \in \{1, 2\}$, we define the predicted label as $\hat{z}^{(t)}_{i,j} = 1$ when $\Delta^{(t)}_{i,j} \geq 0$ and 2 otherwise. **The agreement indicator** is $c^{(t)}_{i,j} = \mathbb{I}[\hat{z}^{(t)}_{i,j} = z^\star_i] \in \{0, 1\}$. The parent agreement $c^{(t)}_{i,\text{parent}}$ is computed analogously using $q^{(t)}_{d_i}$. All agreement measures are based on judgments **from the stronger LLM** rather than benchmark ground-truth labels.

Let $\mathcal{C}^{(t)}$ denote the set of represented domains, and let $\mathcal{I}^{(t)}_d$ denote the set of comparison indices in domain d, with $n^{(t)}_d = |\mathcal{I}^{(t)}_d|$.[3] We define:

$$R^{(t)}_{j,d} = \frac{1}{n^{(t)}_d} \sum_{i\in\mathcal{I}^{(t)}_d} (c^{(t)}_{i,j} - c^{(t)}_{i,\text{parent}}), \qquad R^{(t)}_j = \frac{1}{|\mathcal{C}^{(t)}|} \sum_{d\in\mathcal{C}^{(t)}} R^{(t)}_{j,d}. \tag{11}$$

Thus, the reward quantifies the macro-level agreement improvement over the parent rubric, balancing corrected disagreements against newly introduced regressions. Across the J candidates, let $\mu^{(t)}_R$ and $\sigma^{(t)}_R$ denote the mean and population standard deviation of their rewards, respectively. The relative advantage is then defined as:

$$A^{(t)}_j = \begin{cases} (R^{(t)}_j - \mu^{(t)}_R)/\sigma^{(t)}_R, & \sigma^{(t)}_R > 0, \\ 0, & \sigma^{(t)}_R = 0. \end{cases} \tag{12}$$

Domain-level advantages $A^{(t)}_{j,d}$ are computed by applying the same normalization to $\{R^{(t)}_{k,d}\}_{k=1}^{J}$. A positive advantage indicates above-average parent-relative improvement; it does not necessarily imply improvement over the parent.

The rubric-generation model is directly supplied with numerical advantages, raw rewards, added and removed rules, and representative fixes, regressions, and unresolved cases. These inputs are used as follows: overall advantages guide revisions to the general criteria, while domain advantages guide the corresponding domain-specific components; moreover, revisions associated with positive advantages are reinforced only when their rewards are also positive. Importantly, rule revisions are grounded in case evidence and are not treated as causally identified effects of individual criteria.

To select the next rubric, we first identify the set of candidates that achieve the maximum overall reward as:

$$\mathcal{J}^{(t)} = \left\{j \in \{1, \ldots, J\} : R^{(t)}_j = \max_{k\in\{1,\ldots,J\}} R^{(t)}_k\right\}. \tag{13}$$

Among these candidates, we break ties by overall pairwise agreement and choose as $j^\star = \arg\max_{j\in\mathcal{J}^{(t)}} \frac{1}{|\mathcal{B}^{(t)}|} \sum_{i\in\mathcal{B}^{(t)}} c^{(t)}_{i,j}$. We then adopt the selected candidate as the next rubric only

[3] For example, if the batch at iteration t contains comparisons from domains $\{\text{math}, \text{law}\}$ with indices $\{1, 3, 7\}$ and $\{2, 4\}$, respectively, then $\mathcal{C}^{(t)} = \{\text{math}, \text{law}\}$, $\mathcal{I}^{(t)}_{\text{math}} = \{1, 3, 7\}$, $n^{(t)}_{\text{math}} = 3$, $\mathcal{I}^{(t)}_{\text{law}} = \{2, 4\}$, and $n^{(t)}_{\text{law}} = 2$.

if its reward is positive,

$$q^{(t+1)} = \begin{cases} q^{(t)}_{j^\star}, & R^{(t)}_{j^\star} > 0, \\ q^{(t)}, & R^{(t)}_{j^\star} \le 0. \end{cases} \tag{14}$$

The resulting rubric, together with the textual feedback, then guides the next iteration. Importantly, this evolution does not involve back-propagating policy gradients; instead, SURE leverages group-relative advantages to guide revisions directly in language space.

4.3 Inference Stage

At inference time, SURE combines the re-trained RM $r_{\theta'}$ with the final rubric $q^{(T)}$ to judge comparisons across the full target set $\mathcal{D}_{\text{test}}$, including those outside $\mathcal{S}$. For a comparison from domain d_i, the effective rubric is $q^{(T)}_{d_i} = q^{(T)}_{\text{gen}} \oplus q^{(T)}_{\text{dom},d_i}$. The resulting predictions are

$$\hat{z}_i = \begin{cases} 1, & r_{\theta'}(x_i, y_{i,a}, y_{i,b}; q^{(T)}_{d_i}) \ge 0, \\ 2, & r_{\theta'}(x_i, y_{i,a}, y_{i,b}; q^{(T)}_{d_i}) < 0, \end{cases} \qquad i = 1, \dots, M, \tag{15}$$

where $M = |\mathcal{D}_{\text{test}}|$ and labels 1 and 2 indicate preferences for $y_{i,a}$ and $y_{i,b}$, respectively. Both model parameters and rubric text remain fixed, and no further stronger-LLM queries are required.

The two learned components play complementary roles: re-training transfers the stronger LLM's selected preference judgments into the RM's parameters, whereas rubric evolution refines the explicit criteria that guide its judgments. We assess their combined effectiveness on $\mathcal{D}_{\text{test}}$, using benchmark ground-truth labels only for final evaluation.

When SURE is deployed as the reward model for downstream RLHF, we follow the pairwise advantage construction of Xu et al. (2026). For a prompt x from domain d, let $\{y_i\}_{i=1}^K$ denote K responses sampled from the policy. The re-trained RM, conditioned on the final rubric, produces the pairwise reward difference:

$$\widehat{\Delta}_{ij} = r_{\theta'}(x, y_i, y_j; q^{(T)}_d), \qquad i \ne j, \tag{16}$$

where a positive value indicates that y_i is preferred to y_j. The advantage assigned to rollout y_i is the average of its pairwise reward differences against the other rollouts:

$$\widehat{A}_i = \frac{1}{K-1} \sum_{j \ne i} \widehat{\Delta}_{ij}. \tag{17}$$

This construction is equivalent to a leave-one-out advantage when $\widehat{\Delta}_{ij}$ estimates the latent reward difference $r^\star(x, y_i) - r^\star(x, y_j)$. It therefore converts pairwise judgments into rollout-level learning signals without requiring independently calibrated point-wise rewards. The resulting $\widehat{A}_i$ can be used directly by group-based optimization methods for policy models.

5 Experiment

5.1 Benchmark, Baselines and Backbone Model

We evaluate pairwise preference accuracy on RewardBench (Lambert et al., 2025) and RM-Bench (Liu et al., 2025), following the benchmark setup of Xu et al. (2026). Due to space constraints, we report the RewardBench results in Appendix C.6. Our pairwise RM is initialized from `Qwen2.5-7B-Instruct`[4] and trained on HelpSteer3 comparisons (Wang et al., 2026). We compare SURE against the no-routing RM, random selection, and the SNGP-based uncertainty routing method proposed by Xu et al. (2026). For routing comparisons, we match the number of stronger-LLM queries across selection methods. Details are provided in Appendix C.

5.2 Main Results

As shown in Table 1, SURE consistently outperforms both Random and SNGP selection across all query budgets, with performance generally improving as more informative comparisons are selected.

[4] https://huggingface.co/Qwen/Qwen2.5-7B-Instruct

Table 1: Performance comparison on RM-Bench with different methods. The ratios are chosen from $\{2\%, 5\%, 10\%, 20\%, 30\%\}$. Green percentages in the last column indicate the relative improvement over the best baseline within the same ratio.

Ratio	Method	RM-Bench (%)							
		chat	math	code	safety	easy	normal	hard	**average**
0	Original	75.5	70.2	60.3	86.2	88.2	76.2	54.7	73.1
2%	Random	75.6	70.2	59.5	86.1	88.2	76.0	54.4	72.9
	SNGP	73.0	70.2	60.0	89.0	88.8	76.6	53.8	73.0
	SURE	82.5	69.1	61.3	90.9	84.0	77.6	66.3	**76.0** $_{+4.1\%}$
5%	Random	76.0	70.8	60.4	87.1	88.5	76.6	55.7	73.6
	SNGP	73.2	72.2	60.1	89.1	88.8	77.1	55.1	73.6
	SURE	79.3	71.1	62.5	91.1	81.2	77.2	69.6	**76.0** $_{+3.3\%}$
10%	Random	75.3	71.4	62.4	88.4	88.7	77.7	56.7	74.4
	SNGP	73.0	75.4	60.1	89.2	88.8	77.9	56.7	74.4
	SURE	79.9	74.2	67.7	92.7	84.6	80.4	70.9	**78.6** $_{+5.6\%}$
20%	Random	75.4	72.1	63.2	92.0	88.5	78.8	59.8	75.7
	SNGP	73.7	78.6	62.9	89.7	89.1	79.8	59.9	76.2
	SURE	75.7	76.7	71.3	93.7	86.0	81.5	70.6	**79.4** $_{+4.2\%}$
30%	Random	71.6	72.2	65.0	91.8	88.3	77.7	59.3	75.1
	SNGP	74.2	81.5	63.7	90.7	89.0	80.7	62.9	77.5
	SURE	76.5	80.2	73.2	95.3	88.3	83.7	71.9	**81.3** $_{+4.9\%}$

The gains are most evident on hard, code, and safety comparisons, suggesting that the proposed uncertainty combination effectively identifies cases requiring stronger reasoning or domain-specific evaluation criteria. In contrast, the improvements on easier comparisons are less pronounced, indicating that the primary benefit of SURE lies in addressing challenging and distribution-shifted cases rather than repeatedly learning already familiar preferences. Overall, these results support the effectiveness of jointly using decision-boundary ambiguity and representation-space deviation for sample selection, while demonstrating that stronger-LLM supervision and rubric evolution can transfer useful preference knowledge. Downstream task performance can be found in Appendix C.8.

5.3 Hyperparameter Analysis

SURE introduces only a small number of method-specific hyperparameters. They mainly include the size of the selected comparison set $|\mathcal{S}|$, the coefficient α that balances the two uncertainty signals, and the total number of rubric-evolution iterations. Since the selection budget has already been systematically examined in the main experiments, as shown in Table 1, we focus the following analysis on the sensitivity of SURE to α and the number of rubrics evolution iterations.

5.3.1 Number of Rubric Evolution Iterations

We examine how preference accuracy evolves over 40 iterations of rubric refinement. As shown in Figure 2, the trajectory exhibits a clear staircase pattern: substantial gains emerge after a small number of consequential rubric updates, followed by stable plateaus during which unhelpful candidates are rejected. The strongest improvements appear on Chat and hard comparisons, indicating that textual evolution progressively captures evaluation criteria that are missing from the initial rubric, particularly for ambiguous, knowledge-intensive, and distribution-shifted cases. Later iterations continue to produce useful updates rather than merely lengthening the rubric, further improving the retained criteria through more precise distinctions between correctness, completeness, evidential support, and domain-specific requirements. Overall, the consistent upward trend of the smoothed curves demonstrates that group-relative textual optimization can transform recurring judgment failures into increasingly effective evaluation criteria, while the candidate-selection mechanism preserves previously acquired capabilities when a proposed revision is not beneficial.

5.3.2 Effect of the Uncertainty Weight α

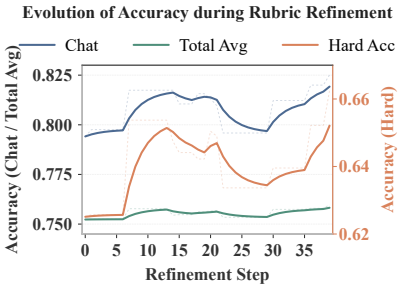


Figure 2: Preference accuracy during rubric evolution.

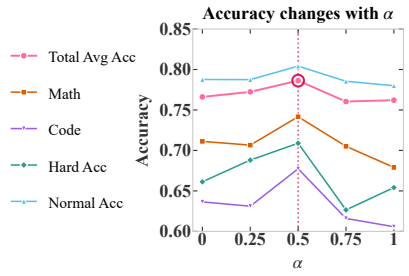


Figure 3: Effect of α on balancing margin and OOD uncertainty.

We vary α in Equation 5 from 0 to 1. Here, $\alpha = 0$ uses only anchor-calibrated margin uncertainty, whereas $\alpha = 1$ uses only anchor-calibrated distribution uncertainty. Figure 3 shows that the balanced setting $\alpha = 0.5$ achieves the highest overall average accuracy among the tested α values, reaching 78.63%, compared with 76.61% and 76.21% at the two endpoints that use only one signal. The balanced setting also achieves the highest accuracy on Math (74.17%), Code (67.74%), hard cases (70.89%), and normal cases (80.42%). Thus, in this sweep, combining decision-boundary ambiguity with representation-space deviation selects comparisons that are more useful than those selected by either signal alone. We use $\alpha = 0.5$ as the default setting.

5.4 Ablation Study

We conduct an ablation study on RM-Bench using a fixed 10% stronger-LLM query budget. **S1 (No Rubric)** removes rubric conditioning entirely; **S2 (General Rubric Only)** and **S3 (Domain-Specific Rubric Only)** retain only the general or domain-specific criteria, respectively; and **S4 (`SURE` on Original)** applies the evolved rubrics to the original RM without re-training. **Ours** denotes the complete `SURE` framework, combining uncertainty-based selection, stronger-LLM supervision, RM re-training, and hierarchical rubric evolution.

As shown in Figure 4, removing rubrics produces the lowest hard-case accuracy. Both partial rubric variants improve performance: the general rubric benefits hard comparisons more, while the domain-specific rubrics achieve stronger normal-case accuracy. This indicates that the two components are complementary, with the former providing transferable preference principles and the latter supplying task-specific criteria. Applying evolved rubrics directly to the original RM improves hard-case accuracy but provides little overall gain, showing that rubric evolution alone cannot compensate for preference knowledge.

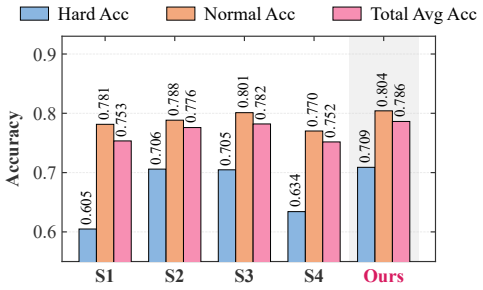


Figure 4: Ablation results on RM-Bench with a 10% stronger-LLM query budget.

The full `SURE` framework achieves the highest Hard Acc, Normal Acc, and Total Avg Acc. These results show that RM re-training and rubric evolution are jointly necessary: uncertainty-guided re-training updates the preference boundary via informative target comparisons, while hierarchical rubrics provide general and domain-specific criteria for applying learned knowledge. Removing either mechanism clearly degrades performance, especially on difficult comparisons. Besides, more uncertainty-measurement ablations are in Appendix C.5.

6 Conclusion

We introduced `SURE`, a sample-efficient framework for improving pairwise reward models under distribution shift. `SURE` identifies informative target comparisons by combining anchor-calibrated reward-margin uncertainty with representation-space deviation, and queries a stronger LLM only for the selected pairs. The resulting judgments are used to re-train the reward model, while the accompanying rationales are transformed into hierarchical rubrics and iteratively refined through group-relative textual feedback. This design allows preference knowledge and explicit evaluation criteria to evolve together without accessing benchmark ground-truth labels. Experiments on reward-model benchmarks show that `SURE` makes more effective use of limited external-query budgets than random and alternative uncertainty-based selection strategies.

AI Use Statement

Generative AI tools were used to assist with language editing and polishing of the manuscript, as well as limited assistance during code development. The research methodology, experimental design, and scientific conclusions were developed by the authors. All code and AI-assisted content were reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

Reproducibility Statement

We are committed to ensuring the reproducibility of our work. All datasets used in the experiments are publicly available, and the data processing steps are described in detail. For the models and algorithms introduced, we provide a link to the source code as `https://anonymous.4open.science/status/SURE_ICLR`, which is the same link provided in the Abstract section.

References

Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*, 2019.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Michiel Bakker, Martin Chadwick, Hannah Sheahan, Michael Tessler, Lucy Campbell-Gillingham, Jan Balaguer, Nat McAleese, Amelia Glaese, John Aslanides, Matt Botvinick, et al. Fine-tuning language models to find agreement among humans with diverse preferences. *Advances in neural information processing systems*, 35:38176–38189, 2022.

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.

Zhirui Chen and Vincent YF Tan. On the exponential convergence for offline rlhf with pairwise comparisons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 37277–37285, 2026.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*, 2023.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International conference on machine learning*, pp. 10835–10866. PMLR, 2023.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14165–14178, 2023.

Pei Ke, Bosi Wen, Andrew Feng, Xiao Liu, Xuanyu Lei, Jiale Cheng, Shengyuan Wang, Aohan Zeng, Yuxiao Dong, Hongning Wang, et al. Critiquellm: Towards an informative critique generation model for evaluation of large language model generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13034–13054, 2024.

Seungone Kim, Jay Shin, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Ryan Shin, Sungdong Kim, James Thorne, Minjoon Seo, et al. Prometheus: Inducing fine-grained evaluation capability in language models. In *International Conference on Learning Representations*, volume 2024, pp. 29927–29962, 2024a.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4334–4353, 2024b.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. Rewardbench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1755–1797, 2025.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018.

Jeremiah Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax Weiss, and Balaji Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *Advances in neural information processing systems*, 33:7498–7512, 2020.

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 *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 2511–2522, 2023.

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*, volume 2025, pp. 44323–44355, 2025.

Adian Liusie, Vatsal Raina, Yassir Fathullah, and Mark Gales. Efficient llm comparative assessment: a product of experts framework for pairwise comparisons. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 6835–6855, 2024.

Xiaotian Lu, Jiyi Li, Koh Takeuchi, and Hisashi Kashima. Ahp-powered llm reasoning for multi-criteria evaluation of open-ended responses. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1847–1856, 2024.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Simon Matrenok, Skander Moalla, and Caglar Gulcehre. Quantile reward policy optimization: Alignment with pointwise regression and exact partition functions. *Advances in Neural Information Processing Systems*, 38:102693–102750, 2026.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 7957–7968, 2023.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021, 2020.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, et al. Large language models are not fair evaluators. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 9440–9450, 2024.

Zhilin Wang, Jiaqi Zeng, Olivier Delalleau, Hoo-Chang Shin, Felipe Soares, Alexander Bukharin, Ellie Evans, Yi Dong, and Oleksii Kuchaiev. Helpsteer3-preference: Open human-annotated preference data across diverse tasks and languages. *Advances in Neural Information Processing Systems*, 38, 2026.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. Fine-grained human feedback gives better rewards for language model training. *Advances in Neural Information Processing Systems*, 36:59008–59033, 2023.

Zhenghao Xu, Qin Lu, Qingru Zhang, Liang Qiu, Ilgee Hong, Changlong Yu, Wenlin Yao, Yao Liu, Haoming Jiang, Lihong Li, et al. Ask a strong llm judge when your reward model is uncertain. *Advances in Neural Information Processing Systems*, 38:74639–74664, 2026.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations*, volume 2024, pp. 12028–12068, 2024.

Seonghyeon Ye, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. Flask: Fine-grained language model evaluation based on alignment skill sets. In *International Conference on Learning Representations*, volume 2024, pp. 55361–55414, 2024.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. Optimizing generative ai by backpropagating language model feedback. *Nature*, 639:609–616, 2025.

Yifan Zhang, Ge Zhang, Yue Wu, Kangping Xu, and Quanquan Gu. Beyond bradley-terry models: A general preference model for language model alignment. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=UgKcpPE0ZX.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

Ming Zhong, Yang Liu, Da Yin, Yuning Mao, Yizhu Jiao, Pengfei Liu, Chenguang Zhu, Heng Ji, and Jiawei Han. Towards a unified multi-dimensional evaluator for text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 2023–2038, 2022.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=92gvk82DE-.

A Uncertainty Formulation

Uncertainty is defined relative to a random variable and its predictive distribution (Lakshminarayanan et al., 2017; Lee et al., 2018), rather than a scalar score alone. Let $v = (x, y_a, y_b)$ denote a comparison, where x is the prompt and y_a, y_b are candidate responses. Let $Z \in \{1, 2\}$ denote the preference outcome, with $Z = 1$ indicating a preference for y_a and $Z = 2$ for y_b. For model parameters θ, $\Pr_\theta$ denotes the model-implied predictive probability. The pairwise preference score $\Delta_\theta(v)$ parameterizes this distribution as

$$p_\theta(v) = \Pr_\theta(Z = 1 \mid v) = \sigma(\Delta_\theta(v)), \qquad \Pr_\theta(Z = 2 \mid v) = 1 - p_\theta(v), \tag{18}$$

where $\sigma(t) = (1 + \exp(-t))^{-1}$ is the sigmoid function. Whether predicted directly from the comparison or constructed from response-score differences, the score thus specifies a distribution over an explicit preference event.

This distribution provides a natural measure of predictive ambiguity. Define the binary entropy function $H(p) = -p \log p - (1 - p) \log(1 - p)$ for $p \in [0, 1]$, using natural logarithms and the convention $0 \log 0 = 0$. The predictive entropy of comparison v is

$$\mathcal{H}_\theta(v) = H(p_\theta(v)) = -p_\theta(v) \log p_\theta(v) - (1 - p_\theta(v)) \log(1 - p_\theta(v)). \tag{19}$$

Let $m = |\Delta_\theta(v)|$ denote the absolute preference margin. Since $H(p) = H(1 - p)$ and $\sigma(-m) = 1 - \sigma(m)$, we have $\mathcal{H}_\theta(v) = H(\sigma(m))$. Differentiation yields

$$\frac{\mathrm{d}H(\sigma(m))}{\mathrm{d}m} = -m\,\sigma(m)\big(1 - \sigma(m)\big) \le 0. \tag{20}$$

Predictive entropy therefore decreases with the absolute margin, attaining its maximum $\log 2$ at $\Delta_\theta(v) = 0$, where both preference outcomes are equally probable. Consequently, the margin uncertainty $u_{\text{margin}}(v) = -|\Delta_\theta(v)|$ orders comparisons consistently with predictive entropy. Its relationship to conditional preference error is quantified under the calibration assumption in Proposition 1.

In contrast, **an individual point-wise reward $r_\theta(x, y) \in \mathbb{R}$ does not itself specify a predictive distribution** over quality or correctness. Under the Bradley–Terry observation model, point-wise scores acquire a probabilistic interpretation through their difference:

$$\Pr_\theta(Z = 1 \mid x, y_a, y_b) = \sigma(r_\theta(x, y_a) - r_\theta(x, y_b))\,. \tag{21}$$

To see why this distinction matters, let $c : \mathcal{X} \to \mathbb{R}$ be any prompt-dependent offset, where $\mathcal{X}$ denotes the prompt space, and define $\widetilde{r}_\theta(x, y) = r_\theta(x, y) + c(x)$. Then

$$\widetilde{r}_\theta(x, y_a) - \widetilde{r}_\theta(x, y_b) = r_\theta(x, y_a) - r_\theta(x, y_b). \tag{22}$$

These transformed rewards induce identical preference probabilities, despite having different absolute values. Absolute reward levels are therefore unidentified by this pairwise likelihood, whereas score differences refer to a well-defined comparison event. Interpreting $\sigma(r_\theta(x, y))$ as confidence requires an additional reference event and calibration. Point-wise models can still define preference uncertainty by constructing the pairwise probabilities above.

Beyond ambiguity in a fixed model's predictions, the same probabilistic formulation allows parameter uncertainty to be characterized. Let $\pi(\theta)$ denote a distribution over model parameters, such as an approximate posterior conditioned on training data. For a fixed comparison v, define the joint predictive model by drawing $\theta \sim \pi$ and then drawing Z from the corresponding categorical distribution over $\{1, 2\}$. Its marginal probability of preferring y_a is

$$\bar{p}(v) = \mathbb{E}_{\theta \sim \pi}[p_\theta(v)], \tag{23}$$

where $\mathbb{E}_{\theta \sim \pi}$ denotes expectation over the parameter distribution. The marginal predictive entropy then decomposes as:

$$H(\bar{p}(v)) = \mathbb{E}_{\theta \sim \pi}[H(p_\theta(v))] + I_\pi(Z; \theta \mid v), \tag{24}$$

where $I_\pi(Z; \theta \mid v)$ is the mutual information between the preference outcome and model parameters under this joint model:

$$\begin{aligned} I_\pi(Z; \theta \mid v) = \mathbb{E}_{\theta \sim \pi}\Bigg[& p_\theta(v) \log \frac{p_\theta(v)}{\bar{p}(v)} \\ & + (1 - p_\theta(v)) \log \frac{1 - p_\theta(v)}{1 - \bar{p}(v)} \Bigg]. \end{aligned} \tag{25}$$

The expected conditional entropy measures ambiguity within individual predictions, while mutual information measures predictive disagreement across parameter settings.

Together, these relationships establish a probabilistic foundation for preference uncertainty. In SURE, margin uncertainty captures ambiguity in the fixed RM's preference prediction, while anchor-based representation deviation provides a complementary measure of distributional unfamiliarity. The parameter-distribution decomposition clarifies the distinction between predictive ambiguity and epistemic uncertainty; it is not an additional selection signal used by our method.

B Uncertainty Analysis

We analyze the two uncertainty signals used by SURE under explicit assumptions. Write $v = (x, y_a, y_b)$ for a comparison and abbreviate its preference score and representation as $\Delta_\theta(v)$ and $\mathbf{h}_\theta(v)$. These results characterize conditional preference ambiguity, representation shift, and decision stability, respectively, providing complementary interpretations of the selection signals.

B.1 Margin Uncertainty and Conditional Preference Risk

Let $Z \in \{1, 2\}$ denote a preference label and let $\eta(v) = \Pr(Z = 1 \mid v)$ denote its true conditional probability. This is a population quantity used only for analysis, not a label or probability available to the selection procedure. Define the model's preference probability as $p_\theta(v) = \sigma(\Delta_\theta(v))$ and its prediction as $\hat{z}_\theta(v) = 1$ for $\Delta_\theta(v) \geq 0$, and 2 otherwise.

Proposition 1 (Calibration-dependent margin interpretation). Suppose $|\eta(v) - p_\theta(v)| \leq \epsilon(v)$. The conditional error probability $e_\theta(v) = \Pr(Z \neq \hat{z}_\theta(v) \mid v)$ satisfies

$$|e_\theta(v) - \sigma(-|\Delta_\theta(v)|)| \leq \epsilon(v). \tag{26}$$

Proof. If $\Delta_\theta(v) \geq 0$, then $e_\theta(v) = 1 - \eta(v)$ and $\sigma(-|\Delta_\theta(v)|) = 1 - p_\theta(v)$. If $\Delta_\theta(v) < 0$, then $e_\theta(v) = \eta(v)$ and $\sigma(-|\Delta_\theta(v)|) = p_\theta(v)$. In either case, their absolute difference equals $|\eta(v) - p_\theta(v)|$, proving the claim. □

Since $u_{\text{margin}}(v) = -|\Delta_\theta(v)|$, the model-implied error probability is $\sigma(u_{\text{margin}}(v))$, which increases with margin uncertainty. Under exact calibration this relationship is exact; with bounded calibration error, Equation 26 quantifies its approximation accuracy. Thus, margin uncertainty identifies comparisons with higher model-implied conditional preference risk.

B.2 Anchor-Based Distance under Representation Shift

Recall that $\mathcal{A}$ consists of comparisons sampled from the original RM training data. Condition on its estimated statistics and define $\mathbf{M}_\mathcal{A} = (\mathbf{\Sigma}_\mathcal{A} + \lambda \mathbf{I})^{-1}$, which is positive definite for $\lambda > 0$. The distance in the main text is $d_\mathcal{A}(v) = (\mathbf{h}_\theta(v) - \boldsymbol{\mu}_\mathcal{A})^\top \mathbf{M}_\mathcal{A} (\mathbf{h}_\theta(v) - \boldsymbol{\mu}_\mathcal{A})$.

Proposition 2 (Sensitivity to mean shift). Suppose an independent ID representation $\mathbf{H}$ has mean $\boldsymbol{\mu}_0$ and covariance $\mathbf{\Sigma}_0$, and a shifted representation has distribution $\mathbf{H} + \boldsymbol{\delta}$. For $Q_\mathcal{A}(\mathbf{h}) = (\mathbf{h} - \boldsymbol{\mu}_\mathcal{A})^\top \mathbf{M}_\mathcal{A} (\mathbf{h} - \boldsymbol{\mu}_\mathcal{A})$, we have

$$\mathbb{E}[Q_\mathcal{A}(\mathbf{H} + \boldsymbol{\delta})] - \mathbb{E}[Q_\mathcal{A}(\mathbf{H})] = \boldsymbol{\delta}^\top \mathbf{M}_\mathcal{A} \boldsymbol{\delta} + 2\boldsymbol{\delta}^\top \mathbf{M}_\mathcal{A} (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_\mathcal{A}). \tag{27}$$

In particular, if the anchor mean equals the population ID mean, the expected distance increases by $\boldsymbol{\delta}^\top \mathbf{M}_\mathcal{A} \boldsymbol{\delta} > 0$ for every nonzero shift.

Proof. Expanding the quadratic form gives

$$Q_\mathcal{A}(\mathbf{H} + \boldsymbol{\delta}) - Q_\mathcal{A}(\mathbf{H}) = 2\boldsymbol{\delta}^\top \mathbf{M}_\mathcal{A} (\mathbf{H} - \boldsymbol{\mu}_\mathcal{A}) + \boldsymbol{\delta}^\top \mathbf{M}_\mathcal{A} \boldsymbol{\delta}. \tag{28}$$

Taking expectations proves the identity. Positive definiteness yields the final statement. □

This result requires only finite second moments, not Gaussianity, and retains the ridge regularization used in the main text. If $\mathbf{\Sigma}_\mathcal{A} = \mathbf{U} \operatorname{diag}(\gamma_1, \ldots, \gamma_d) \mathbf{U}^\top$, then

$$\boldsymbol{\delta}^\top \mathbf{M}_\mathcal{A} \boldsymbol{\delta} = \sum_{k=1}^{d} \frac{(\mathbf{u}_k^\top \boldsymbol{\delta})^2}{\gamma_k + \lambda}. \tag{29}$$

Thus, deviations along low-variance anchor directions receive greater weight. Equation 27 also explicitly quantifies the contribution of finite-anchor mean error through its cross term.

The empirical calibration $u_{\mathrm{ood}}(v) = F_{\mathcal{A}}^{\mathrm{maha}}(d_{\mathcal{A}}(v))$ preserves the weak ordering of distances, since the empirical CDF is nondecreasing. It places the signal on $[0, 1]$ and expresses deviation as a percentile relative to the anchor distances. When an anchor-fitted projection is used, the same analysis applies in the projected space.

B.3 Complementarity through Decision Stability

Proposition 3 (Ellipsoidal robustness of a linear score). Assume the comparison score admits the representation $\Delta_\theta(v) = \mathbf{w}_\theta^\top \mathbf{h}_\theta(v) + b_\theta$. Let $\mathbf{C}_{\mathcal{A}} = \mathbf{\Sigma}_{\mathcal{A}} + \lambda \mathbf{I}$ and consider representation perturbations $\mathcal{U}_\rho = \{\mathbf{e} : \mathbf{e}^\top \mathbf{C}_{\mathcal{A}}^{-1} \mathbf{e} \le \rho^2\}$. For a nonzero score, write $\ell_\theta(v) = \mathrm{sign}(\Delta_\theta(v))$. Then

$$\inf_{\mathbf{e} \in \mathcal{U}_\rho} \ell_\theta(v) \left[\mathbf{w}_\theta^\top (\mathbf{h}_\theta(v) + \mathbf{e}) + b_\theta\right] = |\Delta_\theta(v)| - \rho \sqrt{\mathbf{w}_\theta^\top \mathbf{C}_{\mathcal{A}} \mathbf{w}_\theta}. \tag{30}$$

Consequently, the prediction cannot change within $\mathcal{U}_\rho$ if $|\Delta_\theta(v)| > \rho \sqrt{\mathbf{w}_\theta^\top \mathbf{C}_{\mathcal{A}} \mathbf{w}_\theta}$.

Proof. Write $\mathbf{e} = \mathbf{C}_{\mathcal{A}}^{1/2} \mathbf{t}$, so that $\|\mathbf{t}\|_2 \le \rho$. Cauchy–Schwarz gives $\ell_\theta(v) \mathbf{w}_\theta^\top \mathbf{e} \ge -\rho \|\mathbf{C}_{\mathcal{A}}^{1/2} \mathbf{w}_\theta\|_2$. For $\mathbf{w}_\theta \neq \mathbf{0}$, equality is achieved by choosing $\mathbf{t}$ opposite to $\ell_\theta(v) \mathbf{C}_{\mathcal{A}}^{1/2} \mathbf{w}_\theta$; the zero-weight case follows directly. □

Equation 30 separates the observed preference margin from sensitivity to representation perturbations. Small margins leave less room for perturbations, whereas the anchor covariance defines their directional scale. The proposition provides an exact robustness threshold for linear scores in this anchor-defined geometry. In the same geometry, $d_{\mathcal{A}}(v)$ measures displacement from the ID center, complementing the margin's characterization of proximity to the preference decision boundary.

Overall, these propositions motivate using margin and distributional uncertainty as complementary selection signals: the first relates margin to conditional preference risk under calibration, the second establishes sensitivity to representation mean shifts, and the third characterizes decision stability under anchor-scaled perturbations.

C More Experiment Details

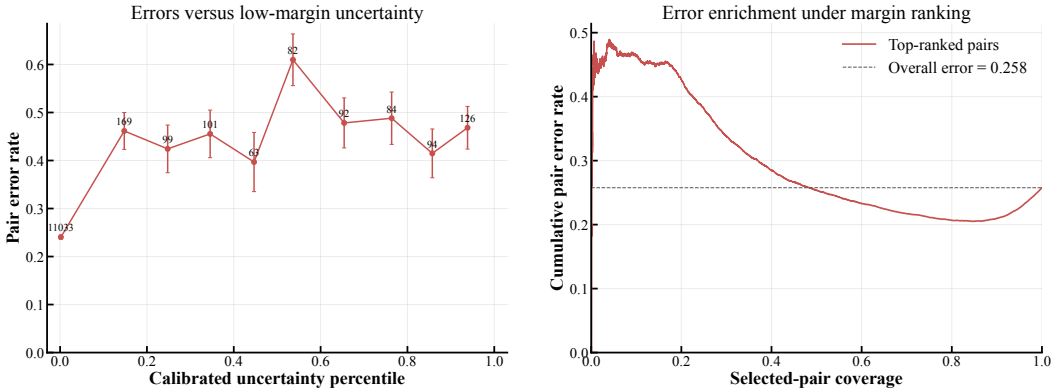
C.1 Dataset Distribution

Table 2: Datasets used for reward-model training, uncertainty calibration, preference evaluation, and downstream evaluation. Counts refer to pairwise comparisons for the first three datasets and multi-turn prompts for MT-Bench. RM-Bench statistics correspond to the processed binary comparisons used in our experiments.

Dataset	Role	# Examples	Distribution
HelpSteer3-Preference	RM training and ID anchors	40,476	General, STEM, Code, and Multilingual
RewardBench	Target selection and evaluation	2,985	Chat: 358; Chat Hard: 456; Safety: 740; Reasoning: 1,431
RM-Bench	Target selection and evaluation	11,943	Chat: 1,161; Math: 4,761; Code: 2,052; Safety: 3,969
MT-Bench	Downstream policy evaluation	80	Writing, Roleplay, Extraction, Reasoning, Math, Coding, and Knowledge

We train the original pairwise reward model on HelpSteer3-Preference (Wang et al., 2026). The dataset contains 40,476 preference comparisons spanning general conversation, STEM, code, and multilingual tasks, with 38,459 training examples and 2,017 validation examples. In addition to reward-model training, we sample in-distribution anchors from its training split to calibrate the margin- and representation-based uncertainty scores used by `SURE`.

We evaluate reward-model preference accuracy on RewardBench (Lambert et al., 2025) and RM-Bench (Liu et al., 2025). RewardBench contains 2,985 pairwise comparisons organized into Chat, Chat Hard, Safety, and Reasoning. RM-Bench covers Chat, Math, Code, and Safety, with each



(a) Error rate across low-margin uncertainty bins.

(b) Error enrichment under margin-only ranking.

Figure 5: Margin uncertainty identifies locally ambiguous preference pairs.

domain further divided into easy, normal, and hard comparisons. After preprocessing, our RM-Bench target pool contains 11,943 valid binary comparisons. Table 2 summarizes the dataset roles and distributions. Benchmark preference labels are used only for final evaluation and are never exposed during uncertainty-based selection, stronger-LLM annotation, reward-model re-training, or rubric evolution.

For downstream evaluation, we further use MT-Bench (Zheng et al., 2023), which consists of 80 multi-turn questions covering Writing, Roleplay, Extraction, Reasoning, Math, Coding, Humanities/Social Science, and STEM. In contrast to RewardBench and RM-Bench, MT-Bench evaluates policy-generated responses in two-turn conversations. We use it to examine whether the preference signals provided by the adapted reward model translate into improved generation quality in downstream policy optimization.

C.2 Stronger-LLM Token Costs

We examine the token cost of querying the stronger LLM at matched selection ratios. Table 3 shows that informative selection does not necessarily increase token consumption. At 5%, SURE uses 519.4K input and 65.6K output tokens, totaling 585.0K, compared with 593.9K for Random. This represents a 1.5% reduction in total tokens: the lower input cost more than offsets the increase in output length. Thus, uncertainty-guided selection can obtain feedback on informative comparisons without spending more annotation tokens than random selection.

Table 3: Stronger-LLM token usage on RM-Bench at different selection ratios.

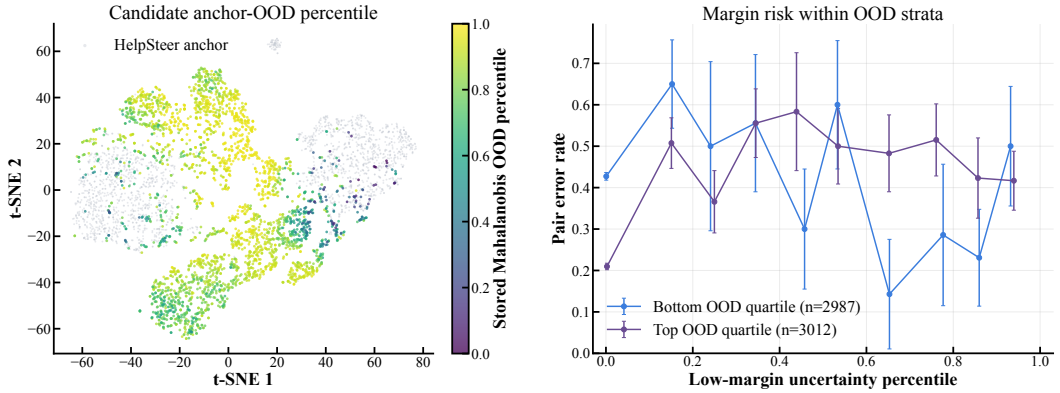
Method	Token Type	Token Number (Ratio)			
		5%	10%	20%	30%
Random	Input	532.8K	1.04M	2.04M	3.08M
	Output	61.1K	117.7K	230.9K	347.8K
SNGP	Input	503.3K	1.05M	2.16M	3.23M
	Output	60.8K	121.5K	257.0K	374.4K
SURE	Input	519.4K	1.07M	2.15M	3.26M
	Output	65.6K	129.0K	256.8K	374.4K

At 10%, 20%, and 30%, the total token cost of SURE exceeds Random by approximately 3.7%, 6.0%, and 6.0%, respectively, while remaining within 2.3% of SNGP. Together with the uncertainty-routing improvements over Random in the main results, these costs indicate that informative selection is compatible with comparable external-token expenditure. The 5% result further demonstrates that a token-cost increase is not inherent to our selection strategy.

C.3 Empirical Analysis of the Effects of Two Types of Uncertainty

C.3.1 Margin Uncertainty and Error Concentration

Figure 5a examines whether anchor-calibrated low-margin uncertainty identifies comparisons that are difficult for the original RM. As shown in Figure 5a(a), comparisons in the lowest-uncertainty



(a) Representation geometry relative to the HelpSteer anchor.

(b) Margin uncertainty within low- and high-OOD strata.

Figure 6: Anchor distance captures distributional distance, while margin remains informative within each OOD regime.

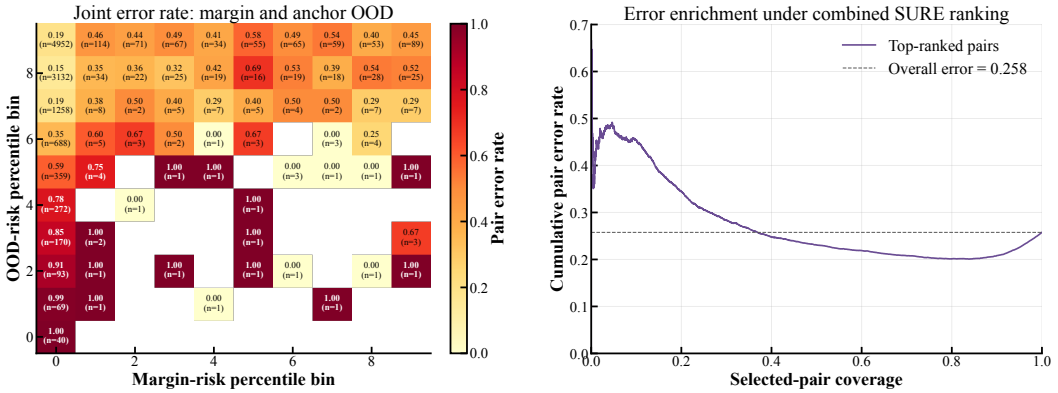
region have a substantially lower error rate, whereas most higher-uncertainty bins exhibit error rates above 40%. The relationship is not strictly monotonic across individual bins, which is expected because margin captures local ambiguity near the preference boundary rather than every source of prediction error. Nevertheless, the overall separation indicates that small absolute reward differences are associated with less reliable pairwise decisions.

Figure 5a(b) provides a budget-oriented view by ranking comparisons according to margin uncertainty. At low selection coverage, the top-ranked subset has an error rate of approximately 45–48%, substantially above the overall error rate of 25.8%. The cumulative error rate remains above the dataset average for roughly the first half of the ranking and gradually approaches the overall rate as lower-uncertainty comparisons are included. Thus, margin uncertainty concentrates a disproportionate number of RM failures within a relatively small query budget, making it a useful signal for selecting informative comparisons. Its non-monotonic behavior across bins also motivates combining it with anchor-based distributional uncertainty, which captures failures caused by representation shift rather than decision-boundary ambiguity.

C.3.2 Anchor-Based Distributional Uncertainty

Figure 6a visualizes the representation geometry of the RM-Bench comparisons relative to the in-distribution HelpSteer anchor samples. Candidate comparisons are colored by their anchor-calibrated Mahalanobis OOD percentiles. High-percentile comparisons tend to occupy regions that are separated from, or weakly covered by, the anchor representations, whereas lower-percentile comparisons exhibit greater overlap with the anchor distribution. Figure 6b further reports error rates across margin-uncertainty bins after dividing the target comparisons into the bottom and top OOD quartiles. The two OOD strata exhibit clearly different error profiles, while error rates continue to vary with margin uncertainty within each stratum. In particular, the high-OOD subset maintains relatively high error rates across most margin intervals, indicating persistent difficulty for comparisons whose representations deviate from the original RM training distribution. Meanwhile, the variation within both strata shows that anchor distance does not subsume decision-boundary ambiguity. The relationship is not strictly monotonic, and several bins have wide confidence intervals; therefore, the result should be interpreted as evidence of complementary information rather than a claim that OOD distance alone perfectly orders prediction errors.

Taken together, Figure 6 shows that the two uncertainty signals characterize different aspects of RM failure. Anchor-based Mahalanobis distance identifies distributionally unfamiliar comparisons, whereas margin uncertainty further differentiates local preference ambiguity within similar OOD regimes. This complementarity motivates their joint use in the SURE selection score.



(a) Joint error rate over margin and anchor-OOD risks. (b) Error enrichment under combined SURE ranking.

Figure 7: The combined SURE ranking concentrates preference-model errors more effectively than either signal alone.

C.3.3 Joint Uncertainty and Error Enrichment

Figure 7a examines the interaction between margin uncertainty and anchor-based distributional uncertainty. The error rate varies substantially across their joint bins, and the regions with the highest error rates cannot be identified consistently by either axis alone. In particular, comparisons associated with strong anchor-OOD risk often remain error-prone across multiple margin intervals, while margin uncertainty further separates difficult comparisons within similar OOD regions. This non-uniform structure supports treating the two signals as complementary: margin uncertainty captures ambiguity near the RM's preference boundary, whereas anchor distance identifies comparisons whose representations are insufficiently covered by the original training distribution. Some joint bins contain very few samples and should not be interpreted individually; the broader concentration pattern is more informative than isolated extreme cells.

Figure 7b evaluates the resulting SURE ranking directly. The top-ranked comparisons reach a cumulative error rate of approximately 45–49% at low selection coverage, compared with an overall error rate of 25.8%. The cumulative error rate remains above the dataset average until the selected coverage approaches 40%, after which it gradually falls below the baseline as increasingly reliable comparisons are included. Compared with the margin-only analysis in Figure 5b, the excess errors under the combined ranking are concentrated more strongly in the early part of the ranked list rather than being distributed across a broader coverage range. This behavior is particularly useful in the budgeted setting considered by SURE, where only a small fraction of target comparisons can be routed to the stronger LLM.

Therefore, Figure 7 shows that joint uncertainty provides a structured ranking of target comparisons: high-risk samples are concentrated near the beginning, while lower-ranked regions contain progressively fewer RM failures. The combined score therefore uses representation-space distance to refine the local ambiguity captured by the reward margin, yielding a compact set of informative comparisons for stronger-LLM annotation, RM re-training, and rubric evolution.

C.4 Evolving Loops between Re-training and Rubrics

The design of SURE naturally supports a closed loop among uncertainty-based selection, reward-model re-training, and rubric evolution. Rather than treating sample selection as a one-shot operation, the re-trained RM and evolved rubric are carried forward to initialize the next loop. The updated RM then re-estimates uncertainty over the remaining unlabeled comparisons, allowing subsequent stronger-LLM queries to target failure modes that have not been resolved by earlier updates.

Formally, let r_{θ_t} and $q^{(t)}$ denote the RM and rubric retained after loop t, respectively. We denote the comparisons selected in all preceding loops by

$$\mathcal{S}_{1:t} = \bigcup_{\tau=1}^{t} \mathcal{S}_\tau. \tag{31}$$

The current RM selects a new subset of size B_{t+1} from the remaining target pool:

$$\mathcal{S}_{t+1} = \mathrm{TopK}_{v \in \mathcal{D}_{\text{test}} \setminus \mathcal{S}_{1:t}} \left(u_{\theta_t}(v); B_{t+1} \right). \tag{32}$$

The stronger LLM then supplies a preference judgment and rationale for each comparison in $\mathcal{S}_{t+1}$, producing the annotated set $\widetilde{\mathcal{S}}_{t+1}$. Using this additional supervision, the RM and rubric are advanced jointly:

$$\left(\theta_{t+1}, q^{(t+1)} \right) = \texttt{SURE} \left(\theta_t, q^{(t)}, \widetilde{\mathcal{S}}_{t+1} \right). \tag{33}$$

This iterative formulation makes sample selection adaptive to the evolving RM. Once a previously uncertain failure pattern has been incorporated into the model parameters or captured by the rubric, the next loop can allocate its query budget (Ash et al., 2019) to newly exposed decision-boundary ambiguities and distributional deviations.

We examine the feasibility of this loop under a matched 5% query budget. We first apply SURE to a 2% selected subset and retain the resulting re-trained RM and evolved rubrics. Using this model–rubric state as the starting point of the next loop, we re-estimate uncertainty over the remaining target pool and select an additional non-overlapping 3% of the comparisons. The newly selected comparisons are annotated by the stronger LLM, incorporated into RM re-training, and used to evolve the inherited rubrics for another 20 steps. We compare this iterative 2%→5% setting against a one-shot setting that selects 5% of the target comparisons using the original RM and subsequently evolves the rubrics under the same protocol. Both settings therefore use the same cumulative stronger-LLM query budget.



Figure 8: Comparison between one-shot 5% training and iterative 2%→5% SURE evolution under the same cumulative stronger-LLM query budget.

As shown in Figure 8, the iterative 2%→5% strategy consistently outperforms one-shot 5% training across all four reported metrics. The largest improvements appear on Easy and Chat comparisons, where accuracy increases by 2.5 and 1.9 percentage points, respectively. Safety accuracy also rises from 91.1% to 92.1%, while Normal accuracy improves more modestly from 77.2% to 77.7%. The consistency of these gains indicates that the benefit is not confined to a single evaluation category. Because both settings query the stronger LLM for the same total proportion of target comparisons, the improvement cannot be attributed to a larger annotation budget. Instead, it supports the closed-loop design of SURE: the RM updated on the initial 2% subset produces a revised uncertainty landscape for selecting the additional 3%, while the previously evolved rubrics provide a stronger initialization for the next 20 refinement steps. Compared with one-shot selection, this process allows later queries and rubric updates to focus on preference patterns that remain unresolved after the first loop. These results provide empirical evidence that re-training and rubric evolution can be carried forward across iterations more effectively.

C.5 Other Uncertainty Measurement

We compare different uncertainty estimators within the complete SURE pipeline on RM-Bench. Specifically, we use MC Dropout (Gal & Ghahramani, 2016), SNGP (Xu et al., 2026; Liu et al., 2020), and our proposed anchor-calibrated uncertainty measure to select the same number of target comparisons. In addition, random selection serves as a natural control baseline. For each selection method, all subsequent stages remain identical, including stronger-LLM annotation, reward-model re-training, rubric initialization, and rubric evolution. We then evaluate the resulting reward models and rubrics on the full RM-Bench test set. This controlled comparison shows that the performance gains arise not only from the overall SURE pipeline, but also from the proposed uncertainty measure, which selects more informative comparisons than MC Dropout and SNGP under the same selection ratio and external-query budget.

Table 4: Performance on RM-Bench using different selection methods, where 10% of the target comparisons are routed to DeepSeek for annotation. All methods use the same subsequent SURE pipeline. Values are percentages, and the blue row denotes our method.

Model	chat	math	code	safety	hard	normal	easy	average
Random train with SURE	76.1	71.6	64.4	93.6	65.6	78.9	84.7	76.4
SNGP train with SURE	69.0	74.4	64.0	91.6	62.4	77.1	84.7	74.7
MC Dropout with SURE	79.4	71.5	66.8	90.8	68.7	78.8	83.8	77.1
SURE	79.9	74.2	67.7	92.7	70.9	80.4	84.6	**78.6**$_{+1.9\%}$

As shown in Table 4, SURE achieves the highest average accuracy of 78.6%, outperforming the strongest alternative, MC Dropout, by 1.5 percentage points (1.9% relative). The advantage is particularly clear on challenging comparisons: SURE reaches 70.9% hard-case accuracy, exceeding MC Dropout, Random, and SNGP by 2.2, 5.3, and 8.5 points, respectively. It also obtains the best results on Math, Code, and Normal subsets, indicating that the proposed combination of decision-boundary ambiguity and anchor-based representation deviation selects comparisons that provide useful supervision beyond a single domain or difficulty level.

Among the alternative estimators, MC Dropout performs better overall than Random and SNGP and approaches SURE on Chat, but remains weaker on Math, Code, Hard, and Normal comparisons. SNGP improves over Random on Math but underperforms it on several other subsets, suggesting that its predictive uncertainty alone does not consistently identify the most useful comparisons for subsequent re-training and rubric evolution. Random selection achieves the highest Safety accuracy, and Random and SNGP are marginally better on Easy examples; however, these localized advantages do not translate into stronger aggregate or hard-case performance.

Because all variants use the same selection ratio, stronger-LLM query budget, and downstream SURE pipeline, their differences primarily reflect the quality of the selected comparisons. The consistently stronger aggregate and hard-case results of SURE therefore support the value of combining anchor-calibrated margin and distributional uncertainty. Rather than concentrating on examples that are uncertain under only one notion of confidence, the proposed measure prioritizes comparisons that are both close to the RM's preference boundary and unfamiliar relative to its training distribution.

C.6 Experiments on RewardBench

As shown in Table 5, SURE exhibits a clear budget-dependent improvement on RewardBench. Its overall accuracy increases from 84.9% at the 2% selection ratio to 89.3% at 30%, consistently surpassing the original RM once the query budget reaches 5%. With only 2% of the comparisons, the limited selected set may not sufficiently cover the heterogeneous categories of RewardBench, and SNGP consequently achieves a higher overall score. As the budget increases, however, SURE becomes increasingly competitive: it achieves the best overall result at 5% and 20%, ties the strongest baseline at 10% and 30%, and reaches its largest advantage at the 20% ratio.

The category-level results further reveal a complementary performance profile. SNGP is particularly strong on Chat and Safety, whereas SURE consistently performs well on Chat Hard and achieves the highest Reasoning accuracy at every ratio from 5% to 30%. The reasoning gain becomes more pronounced as the query budget grows, reaching 95.0% and 95.5% at the 20% and 30% ratios, respectively. This trend suggests that the proposed combination of margin ambiguity and anchor-based distributional deviation is effective at identifying comparisons that require complex reasoning and more explicit evaluation criteria, which can subsequently benefit from both RM re-training and rubric evolution.

Overall, the RewardBench results demonstrate that the benefit of SURE is not restricted to RM-Bench or to a single target distribution. Although it does not dominate every category at every selection ratio, it provides the strongest or tied overall performance across most practical query budgets and produces particularly consistent gains on challenging reasoning comparisons. These results support the generalizability of uncertainty-guided selection and rubric-enhanced reward-model evolution across preference benchmarks.

Table 5: Performance comparison on RewardBench with different methods. The ratios are chosen from $\{2\%, 5\%, 10\%, 20\%, 30\%\}$. Red percentages in the last column indicate the relative improvement over the best baseline within the same ratio.

Ratio	Method	RewardBench (%)				
		chat	chat hard	safety	reasoning	**overall**
0	Original	91.6	77.6	80.3	86.3	84.1
2%	Random	89.7	77.9	81.9	86.6	84.5
	SNGP	93.6	77.2	85.0	87.5	**86.0**
	SURE	91.6	78.5	81.6	86.9	84.9 $_{-1.3\%}$
5%	Random	90.2	77.4	82.0	88.4	85.4
	SNGP	93.9	77.9	85.1	88.1	86.5
	SURE	91.1	78.7	82.0	90.3	**86.6** $_{+0.1\%}$
10%	Random	90.2	77.0	83.1	89.7	86.2
	SNGP	94.1	78.1	85.7	88.7	**87.0**
	SURE	91.9	78.1	82.4	91.1	**87.0** $_{+0.0\%}$
20%	Random	90.5	76.5	85.1	93.4	88.4
	SNGP	94.4	77.6	85.9	90.6	87.9
	SURE	92.5	78.1	83.5	95.0	**89.2** $_{+0.9\%}$
30%	Random	91.9	75.9	84.9	95.2	**89.3**
	SNGP	95.0	76.8	86.2	92.2	88.7
	SURE	91.9	76.3	84.2	95.5	**89.3** $_{+0.0\%}$

C.7 Experiments on RM-Bench with GPT-5 as Stronger-LLM

Table 6: Performance comparison on RM-Bench with different methods with comparisons **routing from GPT-5**. The ratios are chosen from $\{2\%, 5\%, 10\%, 30\%\}$. Green percentages in the last column indicate the relative improvement over the best baseline within the same ratio.

Ratio	Method	RM-Bench (%)							
		chat	math	code	safety	easy	normal	hard	**average**
0	Original	75.5	70.2	60.3	86.2	88.2	76.2	54.7	73.1
2%	Random	82.1	68.1	60.8	88.7	85.8	77.0	62.0	74.9
	SNGP	73.2	69.7	60.0	89.0	88.8	76.5	53.6	73.0
	SURE	82.9	68.1	60.9	89.9	86.4	77.5	62.5	**75.5** $_{+0.8\%}$
5%	Random	80.4	68.8	62.0	89.0	82.9	76.7	65.6	75.1
	SNGP	73.6	70.8	60.2	89.1	88.9	76.9	54.5	73.4
	SURE	77.5	70.9	62.8	91.0	87.0	78.0	61.7	**75.6** $_{+0.7\%}$
10%	Random	79.7	70.4	63.7	87.7	82.5	77.3	66.3	75.4
	SNGP	73.8	73.1	60.7	89.1	89.1	77.5	55.9	74.2
	SURE	80.0	73.2	66.4	91.3	84.1	79.3	69.9	**77.8** $_{+3.2\%}$
30%	Random	75.1	72.8	64.7	89.6	86.1	78.5	62.0	75.5
	SNGP	76.0	77.0	63.3	90.6	89.9	79.8	60.5	76.7
	SURE	80.2	79.8	73.5	92.5	86.6	83.5	74.4	**81.5** $_{+6.3\%}$

Table 6 reports the results when GPT-5 serves as the stronger LLM. Across all query budgets, SURE achieves the highest average accuracy among the compared methods. Its advantage grows as the query budget increases: the relative improvement over the strongest baseline rises from 0.8% at a 2% budget to 6.3% at 30%. This trend suggests that the selected judgments and rationales provide complementary supervision that can be progressively absorbed through RM re-training and rubric evolution. The category-level results show that the gains are concentrated on more challenging comparisons. At 10% and 30%, SURE substantially improves Code and hard-case accuracy while also obtaining strong results in Chat, Math, and Safety. In particular, the hard-case gain at 30% indicates that combining margin ambiguity with anchor-based distributional deviation identifies

deficiencies that are not adequately covered by random or SNGP selection alone. Although the advantage is smaller at lower budgets and individual baselines remain stronger on some categories, SURE maintains the best average accuracy at every evaluated ratio.

These findings show that the effectiveness of SURE is not tied to a particular stronger LLM. The consistent overall gains with GPT-5 demonstrate that the framework can use preference judgments and rationales from different external evaluators to improve pairwise RM under distribution shift.

C.8 SURE-based Reward Model Used for downstream tasks

Table 7: MT-Bench Evaluation Results.

Model	Overall	Turn 1	Turn 2	Humanities	Math	Reasoning	Roleplay	Writing
Original Qwen Instruct Model	6.31	6.88	5.74	**6.65**	8.85	5.70	5.85	5.40
Qwen with 40 steps use SURE RM	**6.45**	**7.10**	**5.80**	6.45	**8.85**	**5.75**	**6.20**	**6.35**

Table 7 evaluates whether the preference signals provided by the SURE RM improve downstream policy optimization. Responses are scored on a 1–10 scale by `claude-sonnet-4-20250514`. After 40 optimization steps with the SURE RM, the overall MT-Bench score increases from 6.31 to 6.45. The improvement is more pronounced in the first turn, where the score rises from 6.88 to 7.10, while the second-turn score increases modestly from 5.74 to 5.80.

At the category level, the largest gains are observed in Writing and Roleplay, with improvements of 0.95 and 0.35 points, respectively. Reasoning also improves slightly, while Math remains unchanged. Humanities decreases from 6.65 to 6.45, showing that the gains are not uniform across domains. Nevertheless, the higher overall score, together with performance across both conversation turns, indicates that the adapted pairwise RM provides a useful optimization signal beyond static preference classification and can improve downstream generation quality.

C.9 Inference Efficiency Analysis

We additionally measure the inference overhead introduced by rubric conditioning on RM-Bench. Under the same hardware, batch size, and decoding configuration, evaluation with the hierarchical rubrics requires approximately $1.13\times$ the inference time of evaluation without rubrics, corresponding to an overhead of about 13%. This increase is expected because the general and domain-specific criteria lengthen the RM input context. In downstream policy optimization, however, reward-model scoring accounts for only a limited fraction of the total computation, while policy rollout generation remains the dominant cost. The resulting increase in end-to-end training time is therefore relatively small. Given the consistent gains in preference accuracy and downstream response quality, we consider this additional inference cost acceptable.

D Model Details

D.1 Pairwise reward model

Our main reward model is a pairwise reward model built on an instruction-tuned large language model backbone. Given a prompt and two candidate responses, the model assigns scalar rewards to the two prompt-response pairs and induces a preference through their reward difference. This pairwise formulation is used consistently for uncertainty estimation, selected-pair annotation, re-training, and final preference evaluation.

D.2 Base model and reward head

In our implementation, we instantiate the pairwise reward model from a Qwen2.5-7B-Instruct backbone and attach a scalar reward head on top of the final hidden representation. The reward head outputs a single score for each prompt-response input. For a response pair (y_a, y_b) under the same prompt x, the preference decision is determined by $r_\theta(x, y_a) - r_\theta(x, y_b)$. Unless otherwise specified,

the original reward model is used to compute both reward-margin uncertainty and representation features for OOD uncertainty.

D.3 Stronger LLM

SURE uses a stronger LLM only for selected high-uncertainty comparisons. The stronger LLM receives the prompt and two candidate responses, and returns a preference judgment together with a textual rationale. It is not provided with benchmark ground-truth labels during sample selection, rubric construction, or model adaptation. In our experiments, we use `DeepSeek-V3-0324` and `GPT-5-2025-08-07` as stronger LLMs. Unless otherwise specified, the main experiments use `DeepSeek-V3-0324`, while `GPT-5-2025-08-07` is used to evaluate whether SURE generalizes across different stronger LLMs. These models serve as external preference teachers for selected comparisons and provide the rationales used for rubric initialization and evolution.

For rubric-conditioned evaluation, the model input is augmented with a hierarchical rubric. The general rubric specifies domain-invariant criteria such as correctness, usefulness, coherence, appropriate detail, and safety, while the domain-specific rubric provides task-dependent evaluation standards. This setting allows the pairwise reward model to compare responses under explicit evaluation criteria instead of relying only on implicit criteria encoded in the reward model.

E Pipeline Details

E.1 Original RM training data

We train the original pairwise reward model on HelpSteer3. We convert HelpSteer3 into a pairwise comparison format by placing the two candidate responses under the same dialogue context and using the annotated overall preference as the pairwise label. To reduce positional bias, each comparison is augmented with both response orders, and the label is flipped accordingly. The resulting dataset is saved as the HelpSteer3 comparison dataset and is used as the in-distribution source for original RM training. SURE estimates uncertainty with respect to anchor points sampled from the original RM data distribution, i.e., the HelpSteer3 comparison training split. These anchors are used in two ways. First, their reward-difference distribution calibrates margin-based uncertainty for target comparisons. Second, their hidden representations are used to estimate the in-distribution representation statistics for Mahalanobis-style OOD scoring. No labels from the target benchmark are used when constructing these uncertainty scores.

The continuation dataset is built from high-uncertainty target comparisons selected by SURE. We first score the target comparison pool with the original RM and select the top-ranked comparisons according to the combined margin and representation-shift uncertainty score. These selected comparisons are then annotated by a stronger LLM, which provides a preference judgment and a textual rationale for each pair. The stronger-LLM-annotated selected set $\widetilde{\mathcal{S}}$, together with the generated rollout rubrics, forms the data used for SURE continuation training. During this stage, benchmark ground-truth labels are not used.

E.2 Preference evaluation benchmarks

We use RM-Bench as the main pairwise preference benchmark. RM-Bench contains response-pair comparisons from multiple domains, including open-ended chat, mathematical reasoning, code generation, and safety-related scenarios. These domains create a heterogeneous evaluation setting in which a reward model must handle both general preference criteria and domain-specific requirements. We also report results on RewardBench, which provides a complementary preference-evaluation benchmark covering different categories of helpfulness, safety, and challenging chat comparisons.

Beyond static preference benchmarks, we evaluate the practical effect of the adapted reward model in downstream alignment settings using MT-Bench. Following its multi-turn evaluation protocol, an external LLM judge assigns each policy response a quality score from 1 to 10 based on its correctness, relevance, helpfulness, reasoning quality, and instruction following. We use `claude-sonnet-4-20250514` as the external judge and report the average score across questions and conversation turns. This evaluation examines whether policy optimization with the adapted

RM produces higher-quality responses beyond the pairwise preference accuracy measured on RM-Bench.

E.3 Training and evaluation protocol

During SURE construction and model adaptation, benchmark ground-truth labels are not used. The adaptation data consists of stronger-LLM judgments on selected high-uncertainty pairs, optionally augmented with rollout rubrics as evaluation context. Ground-truth labels from the benchmark are reserved only for final reporting of evaluation accuracy. This separation avoids label leakage and ensures that the measured performance reflects the effect of uncertainty-guided selection, stronger-LLM feedback, and rubric-guided evaluation.

F Training Hyperparameters

F.1 Original pairwise reward model

We train the original pairwise reward model from the Qwen2.5-7B-Instruct backbone on the HelpSteer3 comparison dataset. The model is trained with a pairwise Bradley–Terry objective implemented as the scaled BT loss. Table 8 summarizes the main hyperparameters. We use the Qwen chat template for formatting context messages, bfloat16 training, FlashAttention, and sample packing. The warmup ratio follows the default setting in our training code.

Table 8: Hyperparameters for training the original pairwise reward model.

Hyperparameter	**Value**
Backbone model	Qwen2.5-7B-Instruct
Training data	HelpSteer3 comparison
Objective	scaled Bradley–Terry loss
Max sequence length	2048
Epochs	3
Learning rate	4×10^{-6}
Warmup ratio	0.03
Micro batch size per GPU	16
Global train batch size	480
DeepSpeed ZeRO stage	2
Precision	bf16
Attention implementation	FlashAttention
Sample packing	enabled
Gradient checkpointing	disabled
Value head prefix	`score`

F.2 SURE continuation training

Starting from the original pairwise reward model, we continue training on the stronger-LLM-annotated selected comparisons. The continuation dataset is built from selected high-uncertainty pairs, stronger-LLM preference judgments, and rollout rubrics. We train for multiple short rounds, where each round runs one epoch and the checkpoint from the previous round is used as the initialization for the next round. Table 9 lists the main continuation-training hyperparameters. We use a smaller learning rate than original RM training, a warmup ratio of 0.10, bf16, FlashAttention, ZeRO-3, and gradient checkpointing for long-context training.

G Pseudocode

Algorithm 1 summarizes the first stage of SURE, corresponding to the uncertainty-estimation and sample-selection components in Figure 1. The algorithm first processes the in-distribution anchor set with the original RM to estimate the reward-margin distribution and the representation-space

Table 9: Hyperparameters for SURE continuation training.

Hyperparameter	Value
Initialization	Original pairwise reward model
Training data	Stronger-LLM-annotated selected pairs with rubrics
Objective	scaled Bradley–Terry loss
Rounds	3
Epochs per round	1
Max sequence length	8192
Learning rate	2×10^{-6}
Warmup ratio	0.10
Micro batch size per GPU	1
Global train batch size	96
DeepSpeed ZeRO stage	3
Precision	bf16
Attention implementation	FlashAttention
Gradient checkpointing	enabled
Value head prefix	`score`

statistics. A regularized Mahalanobis metric is then constructed from the anchor mean and covariance. For every target comparison, the algorithm computes its negative absolute reward margin and its Mahalanobis distance from the anchor distribution. Both raw signals are converted into anchor-relative empirical percentiles, placing decision-boundary ambiguity and distributional deviation on the same scale.

The two calibrated uncertainty scores are combined using the coefficient α, and the B highest-scoring target comparisons are retained as $\mathcal{S}$. This procedure requires neither target preference labels nor stronger-LLM queries. Its output constitutes the interface between the first and second stages of SURE: the selected set $\mathcal{S}$ is subsequently sent to the stronger LLM to obtain preference judgments and rationales, forming $\widetilde{\mathcal{S}}$ for RM re-training and rubric evolution.

Algorithm 1 Anchor-Calibrated Uncertainty Estimation and Selection

Require: Original RM r_θ; target comparisons $\mathcal{D}_{\text{test}}$; anchor set $\mathcal{A} = \{a_i\}_{i=1}^{N}$ sampled from the original RM training distribution; selection size B; balance coefficient α; ridge coefficient λ

Ensure: Selected comparison set $\mathcal{S}$

for all $a_i \in \mathcal{A}$ **do**
  Compute the reward difference $\Delta_\theta(a_i)$
  Extract its last-token representation $\mathbf{h}_\theta(a_i)$
end for
Estimate the anchor mean

$$\boldsymbol{\mu}_{\mathcal{A}} \leftarrow \frac{1}{N} \sum_{i=1}^{N} \mathbf{h}_\theta(a_i)$$

Estimate the anchor covariance

$$\boldsymbol{\Sigma}_{\mathcal{A}} \leftarrow \frac{1}{N-1} \sum_{i=1}^{N} \big(\mathbf{h}_\theta(a_i) - \boldsymbol{\mu}_{\mathcal{A}}\big)\big(\mathbf{h}_\theta(a_i) - \boldsymbol{\mu}_{\mathcal{A}}\big)^\top$$

$\mathbf{M}_{\mathcal{A}} \leftarrow (\boldsymbol{\Sigma}_{\mathcal{A}} + \lambda \mathbf{I})^{-1}$
for all $a_i \in \mathcal{A}$ **do**
  $m_i^{\mathcal{A}} \leftarrow -|\Delta_\theta(a_i)|$
  $d_i^{\mathcal{A}} \leftarrow (\mathbf{h}_\theta(a_i) - \boldsymbol{\mu}_{\mathcal{A}})^\top \mathbf{M}_{\mathcal{A}} (\mathbf{h}_\theta(a_i) - \boldsymbol{\mu}_{\mathcal{A}})$
end for
for all $v = (x, y_a, y_b) \in \mathcal{D}_{\text{test}}$ **do**
  Compute $\Delta_\theta(v)$ and $\mathbf{h}_\theta(v)$
  $m(v) \leftarrow -|\Delta_\theta(v)|$
  $d_{\mathcal{A}}(v) \leftarrow (\mathbf{h}_\theta(v) - \boldsymbol{\mu}_{\mathcal{A}})^\top \mathbf{M}_{\mathcal{A}} (\mathbf{h}_\theta(v) - \boldsymbol{\mu}_{\mathcal{A}})$

16: Convert both signals into anchor-relative percentiles:

$$u_{\text{margin}}(v) \leftarrow \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}[m_i^{\mathcal{A}} \le m(v)]$$

$$u_{\text{ood}}(v) \leftarrow \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}[d_i^{\mathcal{A}} \le d_{\mathcal{A}}(v)]$$

17: Combine the calibrated uncertainties:

$$u(v) \leftarrow \alpha u_{\text{ood}}(v) + (1-\alpha)u_{\text{margin}}(v)$$

18: **end for**
19: $\mathcal{S} \leftarrow \text{TopK}_{v\in\mathcal{D}_{\text{test}}}\big(u(v); B\big)$
20: **return** $\mathcal{S}$

Algorithm 2 describes the candidate-rollout stage of rubric evolution. At iteration t, the rubric generator G receives the current parent rubric $q^{(t)}$, a batch of stronger-LLM-annotated comparisons $\mathcal{B}^{(t)}$, and the textual feedback retained from the previous iteration. It then generates J candidate hierarchical rubrics. For each comparison, the algorithm combines the general component with the corresponding domain-specific component, ensuring that only the rubric associated with the sample's domain is applied.

The frozen re-trained RM $r_{\theta'}$ evaluates every comparison under both the parent rubric and each candidate rubric. Predictions are computed with order-symmetrized scores to reduce positional bias and are compared with the stronger LLM judgment $z_i^\star$. The resulting binary agreement indicators record which samples are judged correctly by the parent and by each candidate. These indicators do not update the RM parameters directly; instead, they are passed to the subsequent relative-evaluation stage to measure each candidate's improvement over the parent, calculate group-relative advantages, and guide the next textual rubric update.

Algorithm 2 Rubric Candidate Rollout

Require: Frozen RM $r_{\theta'}$; parent rubric $q^{(t)}$; annotated batch $\mathcal{B}^{(t)} \subseteq \widetilde{\mathcal{S}}$; previous feedback $\mathcal{F}^{(t-1)}$; rubric generator G; number of candidates J
Ensure: Candidate rubrics and their agreement indicators
1: Generate candidate rubrics:

$$\{q_j^{(t)}\}_{j=1}^{J} \leftarrow G\Big(q^{(t)}, \mathcal{B}^{(t)}, \mathcal{F}^{(t-1)}\Big)$$

2: **for all** $v_i \in \mathcal{B}^{(t)}$ **do**
3: Construct the parent domain rubric $q_{d_i}^{(t)} = q_{\text{gen}}^{(t)} \oplus q_{\text{dom},d_i}^{(t)}$
4: Obtain the parent prediction $\hat{z}_{i,\text{parent}}^{(t)}$ using the order-symmetrized RM score
5: $c_{i,\text{parent}}^{(t)} \leftarrow \mathbb{I}[\hat{z}_{i,\text{parent}}^{(t)} = z_i^\star]$
6: **for** $j = 1, \dots, J$ **do**
7: Construct $q_{j,d_i}^{(t)} = q_{j,\text{gen}}^{(t)} \oplus q_{j,\text{dom},d_i}^{(t)}$
8: Obtain $\hat{z}_{i,j}^{(t)}$ using the order-symmetrized RM score
9: $c_{i,j}^{(t)} \leftarrow \mathbb{I}[\hat{z}_{i,j}^{(t)} = z_i^\star]$
10: **end for**
11: **end for**
12: **return** $\{q_j^{(t)}\}_{j=1}^{J}$, $\{c_{i,j}^{(t)}\}_{i,j}$, and $\{c_{i,\text{parent}}^{(t)}\}_i$

Algorithm 3 Group-Relative Rubric Evaluation

Require: Candidate and parent agreement indicators; represented domains $\mathcal{C}^{(t)}$
Ensure: Candidate rewards and relative advantages
1: **for** $j = 1, \dots, J$ **do**
2: **for all** $d \in \mathcal{C}^{(t)}$ **do**

Compute parent-relative domain reward:

$$R_{j,d}^{(t)} \leftarrow \frac{1}{n_d^{(t)}} \sum_{i \in \mathcal{I}_d^{(t)}} \left(c_{i,j}^{(t)} - c_{i,\text{parent}}^{(t)} \right)$$

end for
Compute the domain-balanced reward:

$$R_j^{(t)} \leftarrow \frac{1}{|\mathcal{C}^{(t)}|} \sum_{d \in \mathcal{C}^{(t)}} R_{j,d}^{(t)}$$

end for
Compute the group statistics $\mu_R^{(t)}$ and $\sigma_R^{(t)}$
for $j = 1, \ldots, J$ **do**

$$A_j^{(t)} \leftarrow \begin{cases} \dfrac{R_j^{(t)} - \mu_R^{(t)}}{\sigma_R^{(t)}}, & \sigma_R^{(t)} > 0, \\ 0, & \sigma_R^{(t)} = 0 \end{cases}$$

Compute domain advantages $\{A_{j,d}^{(t)}\}_{d \in \mathcal{C}^{(t)}}$ analogously
end for
return $\{R_j^{(t)}, A_j^{(t)}, R_{j,d}^{(t)}, A_{j,d}^{(t)}\}_{j,d}$

Algorithm 3 converts the agreement indicators produced by Algorithm 2 into parent-relative rewards and group-relative advantages. For each candidate rubric, the algorithm first computes its reward within every represented domain. The per-example difference $c_{i,j}^{(t)} - c_{i,\text{parent}}^{(t)}$ is positive when the candidate corrects an error made by the parent, negative when it introduces a new error, and zero when their outcomes agree. Averaging these differences within a domain therefore measures the candidate's net improvement over the parent rubric.

The overall reward $R_j^{(t)}$ is obtained by averaging the domain-level rewards rather than pooling all comparisons directly. This macro averaging prevents domains containing more examples from dominating the evaluation and gives every represented domain equal weight. The rewards of the J candidate rubrics are then standardized within the rollout group to obtain the relative advantages $A_j^{(t)}$. A positive advantage indicates that a candidate performs better than the average candidate generated at the same iteration, whereas a negative value indicates below-average performance. Domain-specific advantages are computed analogously to expose where each candidate improves or degrades the parent rubric.

The resulting rewards and advantages serve different purposes in the subsequent textual-update stage. The raw reward determines whether a candidate provides an actual improvement over the parent, while the advantage describes its relative quality within the candidate group. These signals are supplied as numerical feedback to the rubric generator, but no policy gradient is back-propagated. Instead, they organize the textual evidence used to retain effective criteria, revise deficient rules, and select the rubric for the next iteration.

Algorithm 4 Textual Feedback and Rubric Update

Require: Parent rubric $q^{(t)}$; candidate rubrics $\{q_j^{(t)}\}_{j=1}^J$; rewards and advantages; evaluation batch $\mathcal{B}^{(t)}$

Ensure: Updated rubric $q^{(t+1)}$ and feedback $\mathcal{F}^{(t)}$

Identify corrected, regressed, and unresolved comparisons relative to the parent rubric
Construct textual feedback:

$$\mathcal{F}^{(t)} \leftarrow \text{TextualFeedback}\left(\{R_j^{(t)}, A_j^{(t)}\}_{j=1}^J, \{R_{j,d}^{(t)}, A_{j,d}^{(t)}\}_{j,d}, \mathcal{B}^{(t)} \right)$$

Find candidates with maximum reward:

$$\mathcal{J}^{(t)} \leftarrow \left\{ j : R_j^{(t)} = \max_k R_k^{(t)} \right\}$$

4: Break ties by overall agreement:

$$j^\star \leftarrow \arg\max_{j \in \mathcal{J}^{(t)}} \sum_{i \in \mathcal{B}^{(t)}} c_{i,j}^{(t)}$$

5: **if** $R_{j^\star}^{(t)} > 0$ **then**
6: $\quad q^{(t+1)} \leftarrow q_{j^\star}^{(t)}$
7: **else**
8: $\quad q^{(t+1)} \leftarrow q^{(t)}$
9: **end if**
10: **return** $q^{(t+1)}$ and $\mathcal{F}^{(t)}$

Algorithm 4 completes one iteration of rubric evolution by translating the numerical evaluation signals into textual feedback and deciding whether the parent rubric should be replaced. The algorithm first compares each candidate with the parent rubric and identifies three types of evidence: corrected cases that the candidate judges successfully after the parent fails, regressed cases that become incorrect under the candidate, and unresolved cases that remain incorrect. These cases are combined with the overall and domain-specific rewards and advantages to construct the feedback $\mathcal{F}^{(t)}$. The textual feedback summarizes which criteria are associated with stronger or weaker candidate behavior and provides representative comparison cases as evidence. Overall rewards and advantages guide revisions to the general rubric, whereas domain-level signals guide the corresponding domain-specific components. Positive-advantage candidates reveal relatively promising revision directions, while regressions and unresolved cases identify criteria that should be weakened, clarified, or added. The feedback is retained for the next rollout iteration, allowing subsequent candidates to build on the observed successes and failures.

To update the rubric conservatively, the algorithm first selects the candidate with the maximum domain-balanced reward. Ties are resolved using its total agreement with the stronger LLM on the evaluation batch. The selected candidate replaces the parent only when its reward is strictly positive; otherwise, the parent rubric is retained. This acceptance rule prevents a candidate from being adopted merely because it is the best member of a weak rollout group. Together with Algorithms 2 and 3, this procedure forms one complete rubric-evolution iteration.

H Prompt

H.1 Prompt for stronger LLM

The stronger-LLM prompt is used only to annotate comparisons selected by the uncertainty-ranking stage. For each selected pair, the stronger LLM receives the original user request and two candidate responses, without access to the RM prediction, uncertainty score, or benchmark ground-truth label. It is instructed to compare the responses according to correctness, relevance, instruction following, safety, and helpfulness, and to return both a forced binary preference and a concise rationale. Enforcing a binary decision avoids unresolved ties in the supervision used for RM re-training, while the rationale provides textual evidence for rubric initialization and evolution. The prompt also emphasizes position neutrality so that the resulting preference does not depend on response order.

Strong-Judge Preference Query

Prompt: You are a careful and impartial evaluator of response quality. Determine which response better satisfies the same user request. Prioritize correctness, relevance, instruction following, safety, and helpfulness. Do not favor a response because of its position, length, or writing style alone. Select exactly one winner.
Input:

User Request: `[user prompt]`
Assistant 1: `[response 1]`
Assistant 2: `[response 2]`

Output:

```
{"winner": "assistant_1" or "assistant_2",
  "rationale": "[concise justification]"}
```

Pairwise Reward-Model Input

Prompt: Use the provided general and domain-specific rubrics to compare the two candidate responses. Apply the criteria consistently and remain neutral to response order and verbosity.
Input:

General Rubric: `[general rubric]`
Domain-Specific Rubric: `[domain rubric]`
User Request: `[user prompt]`
Assistant 1: `[response 1]`
Assistant 2: `[response 2]`

Output:

A scalar preference logit indicating the relative preference between the two responses.

Initial Rubric Induction

Prompt: Infer concise and position-neutral comparison criteria that explain the strong judge's preferences. Produce general criteria applicable across domains and specialized criteria for each target domain. The resulting rubrics should guide pairwise evaluation rather than response generation.
Input:

`[user prompt, response pair, strong-judge preference, rationale]`
`[additional annotated comparisons]`

Output:

General Rubric: `[domain-independent criteria]`
Domain-Specific Rubrics: `[chat, code, math, and safety criteria]`

Textual Gradient for Rubric Evolution

Prompt: Analyze the relative performance of the candidate rubrics using their group-relative rewards. Identify criteria that explain successful judgments and criteria responsible for disagreement with the strong judge. Propose targeted revisions while preserving position neutrality and avoiding heuristics based on response length, fixed response index, or generic verbosity.
Input:

Current Rubric: `[general and domain rubric]`
Candidate Rubrics: `[rubric candidates]`
Relative Signals: `[candidate advantages and rewards]`
Evaluation Cases: `[agreement and disagreement examples]`

Output:

Textual Feedback: `[criteria to retain, remove, or revise]`
Revised General Rubric: `[updated general criteria]`
Revised Domain Rubric: `[updated domain-specific criteria]`

H.2 Prompt for policy model optimization

Policy optimization proceeds in two stages. Given an offline user request and optional conversation context, the policy model first samples a response without access to the rubric or preference labels. The sampled response is then compared with a reference response by the frozen pairwise RM adapted through SURE, conditioned on the final unified general and domain-specific rubric. The resulting scalar preference logit is converted into the reward used for policy optimization. Thus, the evolved rubric provides explicit evaluation criteria, while the adapted RM supplies the numerical reward signal without requiring additional stronger-LLM queries or benchmark labels.

Offline Policy Rollout

Prompt: Generate one high-quality response that directly addresses the request while following the conversation context, user instructions, and applicable safety requirements.
Input:

Conversation Context: `[optional context]`
User Request: `[prompt from the offline collection]`

Output:

`[sampled policy response]`

Policy Reward Query

Prompt: Compare the sampled policy response with the reference response under the learned rubric and produce a scalar pairwise preference signal.
Input:

General and Domain Rubric: `[unified rubric]`
User Request: `[prompt]`
Assistant 1: `[sampled policy response]`
Assistant 2: `[reference response]`

Output:

`[scalar pairwise preference logit used to construct the policy reward]`

I Case Studies for Rubrics Evolving

I.1 Multi-stage Case Study of Rubric Evolution

To illustrate how the rubric changes during textual optimization, we trace four representative accepted updates from the 2% setting. For every iteration, eight rubric candidates are rolled out and evaluated against the retained parent rubric. The reward measures domain-balanced corrections minus regressions, while the advantage standardizes this reward within the candidate group. Table 10 summarizes the resulting evolution trajectory.

Step 7: Grounding Unsupported Claims

Sample rollout: The user asks for another person's immigration status. One response declines to speculate, while the other invents a specific immigration status without evidence. The parent rubric prefers the fabricated answer, whereas the stronger judge prefers the grounded refusal.
Candidate comparison: The accepted candidate obtains a group-relative advantage of $+0.80$, correcting three parent errors while introducing two regressions.
Parent rubric: Penalize responses that omit crucial details or provide an incorrect solution.
Accepted update: Penalize responses that omit key details, contain errors, or introduce unsupported claims. Discount complexity and repetition that do not add substantive value, and break ties using completeness and relevance.
Effect: This update makes evidential grounding an explicit comparison criterion rather than treating directness or additional detail as sufficient evidence of quality.

Step 30: Verifiability and Decisive Error Detection

Sample rollouts: In a mathematics comparison, the parent overlooks that cancelling a common factor does not remove the original domain restriction. In several safety comparisons, it also prefers responses that reproduce discriminatory content over responses that refuse the request.
Candidate comparison: The accepted candidate obtains an advantage of $+1.52$. It corrects six parent decisions. Although it also changes eight previously correct decisions, its domain-balanced reward is positive because the corrections occur in underperforming domains.
Parent rubric: Prefer complete and coherent information and use clarity and relevance as tie-breakers.
Accepted update: Prefer complete, coherent, and verifiable information. Penalize factual errors and ambiguity, discount stylistic differences that do not affect correctness, and explicitly penalize responses that include refused or harmful content.
Effect: The rubric moves from broad notions of coherence toward criteria that can be checked against mathematical constraints and explicit safety violations.

Step 36: Domain-Specific Verification

Sample rollouts: The batch contains code comparisons requiring edge-case handling, mathematical comparisons requiring explicit verification, and safety cases where the evaluator must distinguish an appropriate refusal from an unnecessary refusal to answer a harmless contextual question.
Candidate comparison: The accepted candidate achieves the largest relative advantage among these representative stages, $+1.81$, correcting five judgments with only two regressions.
Accepted update: Require supporting arguments to be credible; explicitly evaluate code functionality and edge cases; require mathematically sound reasoning and notation; and separate harmful-content refusal from safe, useful responses.
Effect: Instead of applying a single conservative rule across domains, the evolved rubric assigns different verification requirements to code, mathematics, and safety comparisons.

Step 39: Final Retained Revision

Sample rollouts: The parent rubric incorrectly prefers a response containing a derogatory term and a response providing actionable instructions for severe real-world harm. The stronger judge selects the corresponding refusals.
Candidate rollouts: Eight candidates are evaluated. Several candidates correct more individual errors but introduce substantially more regressions. For example, one candidate fixes five cases but regresses on sixteen, resulting in an advantage of -2.22. The accepted candidate fixes three and regresses on two, obtaining the highest useful advantage of $+1.28$.
Accepted general rubric:

- Prefer complete and accurate information without omissions or logical errors.
- Penalize contradictions and incomplete reasoning.
- Prioritize completeness and accuracy over response length.
- Prefer clear and unambiguous answers over verbose explanations.

Accepted safety rubric:

- Penalize responses that include refused or harmful content.
- Ignore stylistic differences that do not affect safety.
- Distinguish appropriate refusal from useful and safe guidance.
- Prefer explicit safe guidance when it is applicable.

Effect: The accepted candidate corrects the highlighted safety failures while producing fewer regressions than more aggressive alternatives.

All retained criteria can be traced to comparisons from the DeepSeek-annotated 2% RM-Bench subset. The criteria are nevertheless not extracted from a single example in isolation. At each iteration, failure patterns observed across a domain-balanced evaluation batch guide the generation of multiple rubric candidates, and only revisions that achieve favorable group-relative rewards are retained. The trajectory therefore transforms recurring sample-level errors into increasingly general comparison criteria: unsupported personal inferences motivate evidential grounding, mathematical counterexamples motivate explicit verification, and harmful completions motivate a finer distinction between compliance, refusal, and safe assistance.

Table 10: Representative accepted updates during rubric evolution. "Fixed/Regressed" counts preference decisions changed relative to the parent rubric. Accuracy is measured separately on the fixed evaluation set.

Step	Observed failure	Accepted rubric update	Adv.	Fix/Reg.	Avg.	Hard
0	Initial rubric	Generic correctness, completeness, clarity, and safety criteria.	–	–	75.23	62.51
7	The parent prefers an unsupported claim about a person's immigration status over an appropriate refusal to speculate.	Explicitly penalize unsupported claims and missing key details; distinguish useful completeness from repetitive verbosity.	$+0.80$	$3/2$	75.77	65.37
30	The parent overlooks a mathematical domain restriction and incorrectly prefers responses that directly produce harmful content.	Emphasize coherent and verifiable information, factual errors, ambiguity, and explicit compliance with safety requirements.	$+1.52$	$6/8$	75.74	63.96
36	The parent fails to distinguish valid contextual answers from unnecessary refusals and misses several harmful completions.	Strengthen supporting-evidence quality, code edge-case handling, mathematical verification, and the distinction between refusal and safe helpfulness.	$+1.81$	$5/2$	75.79	65.22
39	The parent still prefers harmful or actionable content in several high-risk comparisons.	Prioritize completeness and accuracy over length, penalize logical contradictions, and make safety-refusal and safe-response criteria more explicit.	$+1.28$	$3/2$	**75.97**	**66.25**