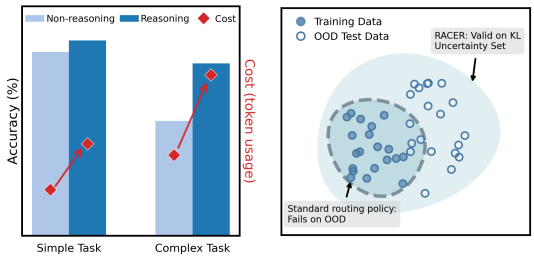
REASONING IS NOT FREE: ROBUST ADAPTIVE COST-EFFICIENT ROUTING FOR LLM-AS-A-JUDGE

Wenbo Zhang*, **Lijinghua Zhang***, **Liner Xiang***, **Hengrui Cai**
Department of Statistics
University of California, Irvine

ABSTRACT

Reasoning-capable large language models (LLMs) have recently been adopted as automated judges, but their benefits and costs in LLM-as-a-Judge settings remain unclear. Through controlled comparisons between reasoning and non-reasoning judges, we show that explicit reasoning substantially improves judgment accuracy on tasks requiring structured verification (e.g., math and coding), while offering limited or even negative gains on simpler evaluations and incurring significantly *higher computational cost*. These findings motivate that reasoning should be used selectively rather than universally, with awareness of possible *distribution shift*. We propose a Robust Adaptive Cost-Efficient Router (RACER), which dynamically selects between reasoning and non-reasoning judges under a fixed budget by formulating routing as a constrained distributionally robust optimization problem. RACER explicitly accounts for distribution shift via a KL-divergence uncertainty set, admits an efficient primal–dual algorithm, and enjoys theoretical guarantees including uniqueness of the optimal policy and linear convergence. Extensive experiments show that RACER achieves superior accuracy–cost trade-offs.

1 INTRODUCTION



(a) Accuracy-cost trade-off **(b)** Robust routing

Figure 1: Motivation of RACER.

As large language models (LLMs) continue to advance, reliably evaluating the quality of their outputs has become increasingly important but challenging. Owing to the high cost and limited scalability of human evaluation, the LLM-as-a-Judge paradigm has emerged as a promising alternative (Zheng et al., 2023; Li et al., 2023; Liu et al., 2023; Kim et al., 2023; Fu et al., 2024), in which LLMs themselves are used as evaluators. In parallel, reasoning-capable LLMs (OpenAI, 2024; Guo et al., 2025) have attracted significant attention. A growing body of work demonstrates that explicitly training models to generate long-form reasoning chains prior to producing final answers substantially improves performance across a wide range of problem-solving tasks, including mathematics, code generation, instruction following, and agentic decision-making (Yang et al., 2025; Liu et al., 2025; Yu et al., 2025; Team et al., 2025). While these findings provide valuable insights into the role of reasoning in LLMs as problem solvers, they do not necessarily imply that such models are effective judges. In current frontier open-source models, reasoning capabilities are typically acquired via training in verifiable problem-solving tasks, whereas the judging ability is not explicitly optimized (Yang et al., 2025; Team et al., 2025; Olmo et al., 2025; Bercovich et al., 2025). This raises a crucial question:*Can reasoning skills learned from problem-solving tasks be transferred to judgment tasks?*

Even if such transfer is possible, deploying reasoning-based judgment models introduces additional challenges. As shown in Fig. 1a, reasoning-based judgments incur substantially *higher computational cost* due to multi-step deliberation and are *not universally beneficial*, as they may induce overthinking

*Equal contribution.

on simple queries and lead to incorrect decisions (Chen et al., 2024b; Sui et al., 2025). A natural mitigation strategy is to learn a routing model that selectively invokes reasoning or non-reasoning models based on criteria such as expected performance or inference cost (Chen et al., 2023; Frick et al., 2025; Liang et al., 2025; Zhang et al., 2025). However, existing routing approaches largely overlook the issues of *distribution shift*. Routers are typically trained on static datasets, whereas real-world queries evolve due to user heterogeneity (Zhang et al., 2024; Chakraborty et al., 2024; Son et al., 2024). Consequently, a router trained under a fixed data distribution may degrade significantly in deployment, resulting in cost constraint violations or erroneous selections of the judging function.

To address these challenges, we propose the **Robust Adaptive Cost-Efficient Router (RACER)**, which formulates router learning as a distributionally robust, constrained policy optimization problem, illustrated in Fig. 1b. The distributionally robust learning framework has recently been used to address the issue of distribution shift in various settings (Namkoong & Duchi, 2016; Rahimian & Mehrotra, 2019; Duchi et al., 2021). This formulation explicitly accounts for distribution shift by optimizing over an uncertainty set of data distributions centered around training data. By solving the resulting robust objective, RACER seeks policies that remain reliable under distribution shift, rather than optimizing solely for average-case performance.

Our **contributions** are summarized as follows:

• We present a comprehensive study comparing reasoning and non-reasoning modes of LLMs in LLM-as-a-Judge settings, characterizing when explicit reasoning improves judgment quality and when it fails, and providing practical insights into the effective use of reasoning for evaluation.
• We formulate router learning in a unified mathematical and algorithmic framework that jointly addresses the cost–performance trade-off and robustness to distribution shift via constrained distributionally robust optimization, followed by a tractable algorithm for efficient solution.
• We are the first in LLM routing to prove that the optimal router policy is unique and our policy iterates converge to it at a linear rate, i.e., the error contracts by a constant factor at each iteration (Wei et al., 2021; Ding et al., 2023). This provides strong theoretical support for the proposed method.
• Extensive experiments demonstrate that RACER consistently outperforms baseline methods under distribution shift, achieving improved accuracy–cost trade-offs.

2 Does Reasoning Help LLM-as-a-Judge?

2.1 Accuracy–Cost Trade-offs of Reasoning Judges

We study whether, and under what conditions, explicit reasoning improves LLM-as-a-Judge. To this end, we perform controlled comparisons between reasoning and non-reasoning judges and characterize the resulting accuracy–cost trade-offs as well as agreement patterns.

Experimental Setup. To isolate the effect of explicit reasoning, we consider paired reasoning and non-reasoning judges $\{\mathcal{M}_{\mathrm{R}}, \mathcal{M}_{\text{no-R}}\}$ and perform controlled comparisons within each pair. By matching the size, architecture, and overall capability of the model, performance differences can be primarily attributed to the presence or absence of reasoning. Concretely, for hybrid models that support both reasoning and non-reasoning inference, we instantiate the pair by treating the reasoning mode as $\mathcal{M}_{\mathrm{R}}$ and the non-reasoning mode as $\mathcal{M}_{\text{no-R}}$. We evaluate multiple open-source hybrid models from the Qwen3 family (1.7B/4B/8B), as well as reasoning and non-reasoning judges on three widely used LLM-as-a-Judge benchmarks: JudgeBench Tan et al. (2024b), RewardBench Lambert et al. (2025), and RewardBench 2 Malik et al. (2025), all of which consist of tasks from various domains. Each example provides a prompt, two candidate responses, and a preference label.

Evaluation Metrics. For evaluation, we prompt the LLM judge to compare the two responses and to provide its preferred choice. We compute *judge accuracy* as the fraction of examples for which the judge's preference matches the ground-truth label, and we record *cost* as token consumption during inference. We define ΔAccuracy as the difference in judgment accuracy between reasoning and non-reasoning modes, $\Delta\text{Accuracy} = \mathrm{Acc}(\mathcal{M}_{\mathrm{R}}) - \mathrm{Acc}(\mathcal{M}_{\text{no-R}})$, and define the cost ratio as the ratio of token consumption under reasoning versus non-reasoning inference.

Results. Fig. 2 (upper) plots ΔAccuracy against the cost ratio, where points closer to the upper-left indicate larger gains at lower additional cost. Across all three benchmarks, the strongest and most cost-effective gains concentrate in math and coding, whereas safety and knowledge show limited

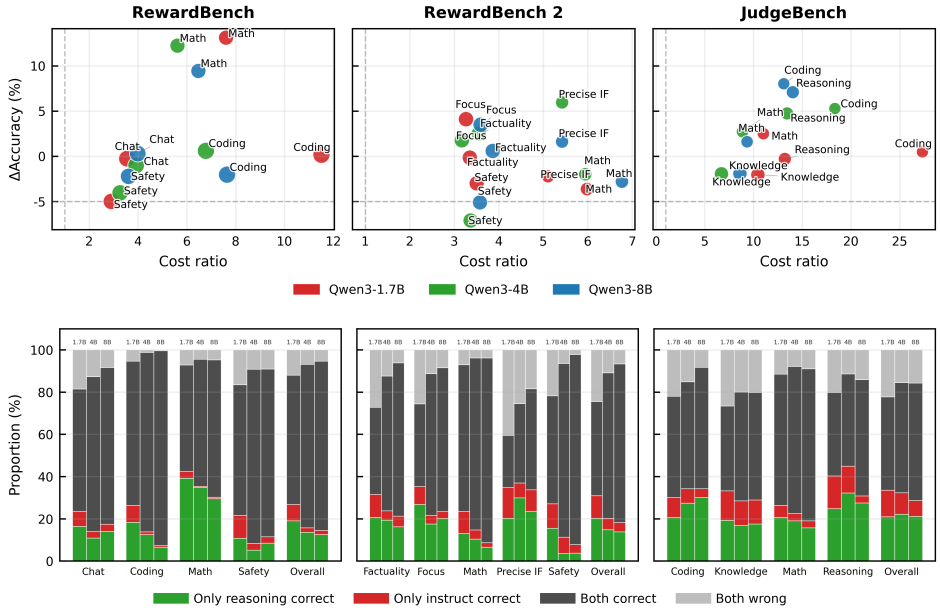


Figure 2: Accuracy–cost trade-offs and reasoning–instructional agreement across benchmarks. **Upper**: Accuracy improvement versus cost ratio. **Lower**: Agreement patterns between instruct and reasoning inference.

improvements and can even degrade under reasoning. Fig. 2 (lower) summarizes agreement patterns between reasoning and non-reasoning judges. When both modes reach the same outcome, non-reasoning is preferable due to its lower cost; when they disagree, selecting the better judge provides additional headroom. This headroom shrinks with model size, as larger non-reasoning judges already achieve stronger baseline judgment accuracy.

Overall, reasoning judge yields significant performance gains, especially in reasoning-intensive domains, illustrating that skills learned from problem-solving tasks effectively transfer to judgment tasks. Also, we notice that the reasoning should be used selectively, given its substantial cost.

Moreover, the heterogeneous accuracy–cost trade-offs in Fig. 2 suggest that both reward and compute cost can shift across domains and benchmarks. As a result, a router tuned on in-distribution data may mis-estimate either the benefit of reasoning or the risk of budget violation under distribution shift. This motivates our distributionally robust objective in Eq. (4), which applies robustness separately to the reward and cost terms to hedge against these two failure modes.

2.2 When and Why Reasoning Helps

To better understand when and why reasoning improves LLM-as-a-Judge, we conduct a case analysis. This analysis reveals three recurring patterns: (i) reasoning improves judgment when evaluation requires explicit verification; (ii) reasoning is largely redundant when the correct choice is apparent from surface cues; and (iii) reasoning can hurt when it over-expands the evaluation scope and introduces irrelevant considerations. Representative examples are summarized in Table D.1.

In math and coding tasks, where correct evaluation often requires checking intermediate steps, validating internal consistency, or reconciling multiple criteria, reasoning-mode judges are better aligned with these requirements. In contrast, for tasks such as factual recall and concise question answering, reasoning and non-reasoning judges usually agree, leaving little room for improvement. Finally, we observe failure cases where explicit reasoning introduces irrelevant factors.

Taken together, these case-level patterns help explain the heterogeneous accuracy–cost trade-offs observed in Section 2.1. They provide qualitative evidence that reasoning improves judgment primarily when its inductive bias aligns with the evaluation structure of the task, reinforcing the conclusion that reasoning should be activated selectively rather than unconditionally under constraints.

3 RACER: ROBUST ADAPTIVE COST-EFFICIENT ROUTER

3.1 NOTATION

We denote the prompt space by $\mathcal{X}$, the response space by $\mathcal{Y}$. Let $\mathcal{Z} = \mathcal{X} \times \mathcal{Y} \times \mathcal{Y}$. We use $\rho \in \Delta(\mathcal{Z} \times \{0, 1\})$ to denote the distributions over pairwise prompt–response tuples with preference labels where $\Delta(\cdot)$ is a probability simplex. For a given pairwise prompt–response tuple (x, y_1, y_2), the label $l = 1$ indicates that y_1 is preferred to y_2, denoted as $y_1 \succ y_2$, while $l = 0$ indicates that $y_2 \succ y_1$. We define the judge function of the model a as $\Phi_a : \mathcal{Z} \mapsto \{0, 1\}$, which maps $z = (x, y_1, y_2)$ to a binary preference label, indicating whether $y_1 \succ y_2$ ($\Phi_a(z) = 1$) or $y_2 \succ y_1$ ($\Phi_a(z) = 0$). Here we assume deterministic judgments for simple cases, but the framework readily extends to stochastic settings. We denote the empirical distribution of the dataset with size n as $\rho_n = \frac{1}{n} \sum_{i=1}^{n} \delta_{(z_i, l_i)}$, where $(z_i, l_i) \sim \rho$.

3.2 PROPOSED METHOD: RACER

Motivated by the accuracy–cost trade-offs of reasoning judges, we propose to learn a router that selectively activates reasoning modes. We formulate router learning as a constrained optimization problem. We define router policy $\pi : \mathcal{Z} \to \Delta(0, 1)$, and denote the sampled decision from $\pi(\cdot \mid z)$ as a taking a value of 1 if choosing the reasoning mode and 0 if choosing the non-reasoning mode. Our optimization objective in terms of the budget C is given by:

$$\max_{\pi \in \Pi} \mathbb{E}_{(z,l)\sim\rho_n, a\sim\pi(\cdot|z)}[r(z, a, l)], \text{ s.t. } \mathbb{E}_{(z,l)\sim\rho_n, a\sim\pi(\cdot|x)}[c(z, a)] \leq C, \tag{1}$$

where $r(z, a, l) = \mathbb{I}(\Phi_a(z) = l)$ with indicator function $\mathbb{I}(\cdot)$ and $c(z, a)$ denotes *whether the judgment is correct* and the *cost* of using the selected mode, respectively, and Π is the policy class of interest. We call this method Adaptive Cost-Efficient Router (ACER). The core idea here is that the routing policy should maximize judgment reward while controlling cost within a budget, yielding an adaptive accuracy–cost trade-off.

However, ACER is not explicitly designed to handle distribution shifts, which are common in real-world scenarios. To address this limitation, we further propose **Robust Adaptive Cost-Efficient Router (RACER)**, which not only optimizes performance under cost constraints, but also remains robust to distribution shifts through distributionally robust learning.

We reformulate the problem as follows:

$$\max_{\pi \in \Pi} \min_{\tilde{\rho} \in \mathcal{U}(\rho_n, \delta)} \mathbb{E}_{(z,l)\sim\tilde{\rho}, a\sim\pi(\cdot|z)}[r(z, a, l)], \text{ s.t. } \max_{\tilde{\rho} \in \mathcal{U}(\rho_n, \delta)} \mathbb{E}_{(z,l)\sim\tilde{\rho}, a\sim\pi(\cdot|z)}\left[c\left(z, a\right)\right] \leq C. \tag{2}$$

This optimization is over a distributional uncertainty set within a distance of at most δ with ρ_n. Here, we utilize the uncertainty set $\mathcal{U}(\rho_n, \delta) = \{\tilde{\rho} : D_{\mathrm{KL}}(\tilde{\rho} \| \rho_n) \leq \delta\}$ and D_{KL} to denote the KL-divergence. Overall, RACER seeks to maximize the worst-case reward under a worst-case cost constraint, explicitly incorporating distributional robustness to prompt and response shifts, an aspect largely overlooked by existing routing methods. We now turn to solving (2).

We define the worst-case reward $R_{\mathcal{U}(\rho_n,\delta)}(\pi)$ and the worst-case cost $C_{\mathcal{U}(\rho_n,\delta)}(\pi)$ as

$$R_{\mathcal{U}(\rho_n,\delta)}(\pi) := \min_{\tilde{\rho} \in \mathcal{U}(\rho_n,\delta)} \mathbb{E}_{(z,l)\sim\tilde{\rho},\, a\sim\pi(\cdot|z)}\big[r(z, a, l)\big], C_{\mathcal{U}(\rho_n,\delta)}(\pi) := \max_{\tilde{\rho} \in \mathcal{U}(\rho_n,\delta)} \mathbb{E}_{(z,l)\sim\tilde{\rho},\, a\sim\pi(\cdot|z)}\big[c(z, a)\big].$$

Hence, the original problem (2) can be rewritten as:

$$\max_{\pi \in \Pi} R_{\mathcal{U}(\rho_n,\delta)}\left(\pi\right) \quad \text{s.t. } C_{\mathcal{U}(\rho_n,\delta)}\left(\pi\right) \leq C. \tag{3}$$

Then the Lagrangian min-max problem associated with (3) is given by:

$$\max_{\pi \in \Pi} \min_{\lambda \geq 0} L(\pi, \lambda) := \max_{\pi \in \Pi} \min_{\lambda \geq 0} R_{\mathcal{U}(\rho_n,\delta)}\left(\pi\right) - \lambda C_{\mathcal{U}(\rho_n,\delta)}\left(\pi\right),$$

where λ is the Lagrange multiplier. While this formulation is solvable through alternating gradient descent methods, it is computationally challenging since we do not have direct control over the data distribution $\tilde{\rho} \in \mathcal{U}(\rho_n, \delta)$ as they are not parameterized distributions. Moreover, the training data are sampled only from the source distribution ρ, without samples from other distributions in the uncertainty set $\mathcal{U}(\rho, \delta)$. To overcome this challenge, we introduce principled tractable algorithms.

The following proposition shows that the worst-case probability distribution within a KL uncertainty set can be efficiently approximated. Similar ideas have appeared in prior works on distributionally robust reinforcement learning (Gadot et al., 2024; Xu et al., 2025).

Theorem 3.1. *Let $\rho_n \in \Delta_n$ be the empirical distribution over $\{z_i, l_i\}_{i=1}^n$, and define $f_i := \mathbb{E}_{a\sim\pi(\cdot|z_i)}[f(z_i, a)]$. Let*

$$\mathcal{U}(\rho_n, \delta) := \Big\{\tilde{\rho} \in \Delta_n : \sum_{i=1}^n \tilde{\rho}(i) \log \frac{\tilde{\rho}(i)}{\rho_n(i)} \le \delta\Big\}.$$

Define $\underline{\rho}$ and $\overline{\rho}$ as solutions to

$$\underline{\rho} \in \arg\min_{\tilde{\rho}\in\mathcal{U}(\rho_n,\delta)} \sum_{i=1}^n \tilde{\rho}(i) f_i,\ \overline{\rho} \in \arg\max_{\tilde{\rho}\in\mathcal{U}(\rho_n,\delta)} \sum_{i=1}^n \tilde{\rho}(i) f_i.$$

Then there exist $\underline{s}, \overline{s} \in \mathbb{R}$ and $\tau > 0$ such that, for all i,

$$\underline{\rho}(i) \propto \rho_n(i) \exp\left(\frac{\underline{s} - f_i}{\tau}\right),\ \overline{\rho}(i) \propto \rho_n(i) \exp\left(\frac{f_i - \overline{s}}{\tau}\right),$$

with $\underline{s} \le \sum_{i=1}^n \rho_n(i) f_i$ and $\overline{s} \ge \sum_{i=1}^n \rho_n(i) f_i$.

We leave the proof of Theorem 3.1 to the Appendix B.1. Based on this Theorem, we get a closed-form solution of $\underline{\rho}, \overline{\rho}$ over KL uncertainty set $\mathcal{U}(\rho_n, \delta)$. The resulting distributions $\underline{\rho}$ and $\overline{\rho}$ are obtained by reweighting the data. In the minimization setting, where f is a reward function, $\underline{\rho}$ downweights samples whose expected reward exceeds the baseline $\underline{s}$ and upweights those with lower-than-baseline rewards. In the maximization setting, where f is a cost function, $\overline{\rho}$ upweights samples with higher-than-baseline cost, focusing optimization on high-risk regions. The parameter τ controls the strength of reweighting: lower τ induces a more aggressive concentration on extreme (worst-case) samples.

Since the baselines $\underline{s}$ and $\overline{s}$ are generally unknown, in practice we approximate them using the empirical mean, which serves as a valid surrogate by providing upper and lower bounds for $\underline{s}$ and $\overline{s}$, respectively. Thus, the Lagrangian min-max problem of (**??**) can be transformed as:

$$\max_{\pi\in\Pi}\min_{\lambda\ge 0} L(\pi,\lambda) = \max_{\pi\in\Pi}\min_{\lambda\ge 0} R_{\mathcal{U}(\rho_n,\delta)}(\pi) - \lambda C_{\mathcal{U}(\rho_n,\delta)}(\pi) = \max_{\pi\in\Pi}\min_{\lambda\ge 0} R_{\underline{\rho}}(\pi) - \lambda C_{\overline{\rho}}(\pi). \quad (4)$$

where the last equality is implied by Theorem 3.1.

We introduce a regularized Lagrangian:

$$L_\beta(\pi,\lambda) := L(\pi,\lambda) + \beta\left(\mathcal{H}(\pi) + \frac{1}{2}\lambda^2\right), \quad (5)$$

where $\mathcal{H}(\pi) + \frac{1}{2}\lambda^2$ is added as a regularization to the original Lagrangian $L(\pi,\lambda)$. Here β is a regularization parameter, and $\mathcal{H}(\pi) := \mathbb{E}_{(z,l)\sim\rho}[\mathcal{H}(\pi(\cdot \mid z)]$ is the entropy of the policy π. This step allows us to control the randomness of the routing policy, thereby encouraging exploration and facilitating convergence (Cen et al., 2022; Ding et al., 2023).

We can then use the primal-dual method to solve (5):

$$\pi_{t+1} = \operatorname*{argmax}_{\pi\in\Pi}\Big\{R_{\underline{\rho}}(\pi) - \lambda_t C_{\bar{\rho}}(\pi) + \beta\mathcal{H}(\pi)\Big\}, \quad (6)$$

$$\lambda_{t+1} = \operatorname*{argmax}_{\lambda\ge 0}\left\{-\lambda_t C_{\bar{\rho}}(\pi) + \frac{1}{2}\beta\lambda_t^2\right\}. \quad (7)$$

Based on the derivations, we propose a practical **RACER** algorithm in Algorithm 1.

4 Theoretical Results

In this section, we first show that the optimization problem admits a unique solution, and then demonstrate linear convergence in terms of the KL divergence between the last iterate and the optimal router policy. While recent works study routing strategies for LLMs with a focus on practical model selection and performance–cost tradeoffs (Ong et al., 2025; Liang et al., 2025; Zhang et al., 2025), these approaches are primarily empirical and heuristic in nature. To the best of our knowledge, this work is the *first* to provide theoretical guarantees on *the convergence behavior of an LLM router*.

Algorithm 1 RACER

input: Number of iterations T, temperature τ, regularization parameter β, distribution ρ for preference data generation, judging functions (Φ_0, Φ_1) by an LLM

1: Initialize π_0 and λ_0.
2: **for** Iteration $t = 0, 1, \ldots, T-1$ **do**
3: **Construct** $\mathcal{D}_t = \{(z, l)\}$ where $(z, l) \sim \rho$.
4: **Sample routing actions:** $a \sim \pi_t(\cdot \mid z), \forall z \in \mathcal{D}_t$ (*optional: enumerate all actions*).
5: **Calculate reward and cost:** For each $(z, l) \in \mathcal{D}_t$ and routing choice a, get judge results $\Phi_a(z)$, and calculate the reward $r(z, a, l)$ and cost $c(z, a)$.
6: **Data reweighting:** Calculate batch mean reward $\overline{r} = \frac{\sum_{j=1}^{|\mathcal{D}_t|} r(z_j, a_j, l_j)}{|\mathcal{D}_t|}$ and cost $\overline{c} = \frac{\sum_{j=1}^{|\mathcal{D}_t|} c(z_j, a_j)}{|\mathcal{D}_t|}$, then approximate worse-case distributions:

$$\underline{\rho}(i) \propto \exp\left(\frac{\overline{r} - r_i}{\tau}\right), \bar{\rho}(i) \propto \exp\left(\frac{c_i - \overline{c}}{\tau}\right).$$

7: **Primal-dual update:** Obtain π_{t+1} and λ_{t+1} by (6) and (7).
8: **end for**
9: **return** The best valid $\pi_{0:T}$ on the validation data.

4.1 Uniqueness of the Optimal Router Policy

The policy optimization problem can be interpreted as finding a saddle point of the max–min problem,

$$\max_{\pi \in \Pi} \min_{\lambda \geq 0} \mathcal{L}_\beta(\pi, \lambda). \tag{8}$$

After establishing feasibility (i.e., the constraint set is nonempty) and showing that the solution space is bounded, we then prove that the saddle point exists and is unique, as stated in Theorem 4.1. All proofs are provided in Appendix B.2.

Theorem 4.1 (Existence and Uniqueness of the Saddle Point)**.** *There exists a unique pair* $(\pi^*, \lambda^*) \in \Pi \times \Lambda$ *such that* $\mathcal{L}_\beta(\pi, \lambda^*) \leq \mathcal{L}_\beta(\pi^*, \lambda^*) \leq \mathcal{L}_\beta(\pi^*, \lambda)$ *for any* $\pi \in \Pi$ *and* $\lambda \in \Lambda$*, that is,* (π^*, λ^*) *is the saddle point of* $\mathcal{L}_\beta(\pi, \lambda)$*.*

As a consequence, for any $(\pi, \lambda) \in \Pi \times \Lambda$,

$$\mathcal{L}(\pi, \lambda^*) - \beta \mathcal{H}(\pi) \leq \mathcal{L}(\pi^*, \lambda^*) \leq \mathcal{L}(\pi^*, \lambda) + \frac{\beta}{2}\lambda^2,$$

which indicates that (π^*, λ^*) is a saddle point of the original Lagrangian $\mathcal{L}(\pi, \lambda)$, up to two β-regularization terms.

Theorem 4.1 identifies a unique target for the primal–dual iterates. In the next subsection, we show that the updates (6) and (7) converge to this unique saddle point.

4.2 Convergence of the Router Policy

We next analyze the last-iterate convergence of our router policy π_t to the optimal policy π^*.

To derive a closed-form solution for (6), we rewrite it as

$$\pi_{t+1} = \arg\max_{\pi \in \Pi} \mathbb{E}_{(z,l)\sim\rho, a\sim\pi(\cdot|z)} \left[\frac{p_{\underline{\rho}}}{p_\rho} r(z, a, l) - \lambda_t \frac{p_{\overline{\rho}}}{p_\rho} c(z, a)\right] + \beta\, \mathbb{E}_{(z,l)\sim\rho} \left[\mathcal{H}\left(\pi(\cdot \mid z)\right)\right],$$

where $p_{\overline{\rho}}$, $p_{\underline{\rho}}$, and p_ρ denote the densities of the distributions $\overline{\rho}$, $\underline{\rho}$, and ρ, respectively.

We then propose Assumptions 1, 2, and 3 to establish the convergence result in Theorem 4.2.

Assumption 1 (Convexity of the policy class Π)**.** The interested policy class Π is convex, i.e., for any $\pi_1, \pi_2 \in \Pi$ and any $\alpha \in [0, 1]$, $\alpha\pi_1 + (1 - \alpha)\pi_2 \in \Pi$.

Assumption 2 (Boundness of c)**.** There exists a constant $M > 0$, such that $c(z, a) \leq M$ for all $z \sim \rho$ and $a \in \{0, 1\}$.

Assumption 3 (Boundness of density ratio)**.** There exists a constant $K > 0$, such that $p_{\overline{\rho}}/p_\rho \leq K$ for all $z \in \mathcal{Z}$.

Assumption 1 is standard and has been widely adopted in convex optimization and policy optimization analyses (see, e.g., Mutti et al., 2023; Ding et al., 2023). Assumption 2 ensures that the cost $c(z, a)$ is bounded and has bounded variance, while Assumption 3 controls the amplification induced by importance reweighting. Together, these conditions imply that the dual gradient term $(p_{\overline{\rho}}/p_\rho)\, c(z, a)$ is uniformly bounded. This uniform bound allows us to control how much the policy π_{t+1} changes in the dual variable λ_t, which is a key ingredient in establishing the linear convergence result below.

Theorem 4.2 (Linear Convergence of RACER)**.** *Under Assumptions 1, 2, and 3, the router policy iterates satisfy*

$$\mathrm{KL}(\pi_t \| \pi^*) \leq \frac{M^2 K^2}{2\beta^2} \left(\frac{M^2 K^2}{M^2 K^2 + 2\beta^2} \right)^{2t} (\lambda_0 - \lambda^*)^2.$$

5 Experiments

5.1 Experimental Setup

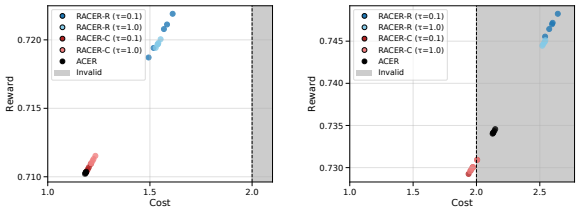
We conduct two complementary sets of experiments with a shared training and judgment generation protocol. The first focuses on controlled ablations designed to isolate the roles of reward and cost robustness under targeted distribution shifts. The second evaluates routing performance on real world benchmarks to assess generalization under more realistic evaluation distributions.

Across both experimental settings, we use a unified data generation protocol to learn a router from the preference data. Given any preference dataset consisting of a prompt and a response pair $(x_i, y_{i,1}, y_{i,2})$, we prompt an LLM judge under two inference modes (instruct and reasoning) to select its preferred response. For each instance, we record (i) whether the judge's decision matches the ground-truth preference label, r_i, and (ii) the number of tokens consumed under each inference mode, c_i. The router takes as input a text embedding of the concatenated context $z_i = (x_i, y_{i,1}, y_{i,2})$ obtained from an embedding model.

Details of the judging prompts, decoding configurations, and the embedding model are provided in Appendix C. All training and evaluation sets in the following experiments are constructed using this protocol, differing only in the underlying dataset.

5.2 Ablating Reward and Cost Robustness

As shown in our robust objective (3), the proposed formulation incorporates two levels of robustness: robustness applied to the reward, denoted as $R_{\mathcal{U}(\rho_n,\delta)}(\pi)$, and robustness applied to the cost, denoted as $C_{\mathcal{U}(\rho_n,\delta)}(\pi)$. To better understand the contribution of each component, we conduct an ablation study to investigate (i) when reward robustness is necessary and (ii) when cost robustness is necessary. Specifically, we introduce two variants, RACER-R and RACER-C, which apply distributionally robust reweighting only to the reward and only to the cost, respectively. We also include ACER with the non-robust objective in (1) as a baseline.



(a) Shift toward **lower-cost queries**.

(b) Shift toward **higher-cost queries**.

Figure 3: Reward–cost trade-offs with a budget of 2.

We consider two representative scenarios: (a) the OOD queries require less computation than in-distribution queries, and (b) the OOD queries require more computation than in-distribution queries. We construct our datasets from the Skywork Reward Preference Dataset Liu et al. (2024a), with Magpie-Ultra and WildGuardMix used for training in Scenarios (a) and (b), respectively, and

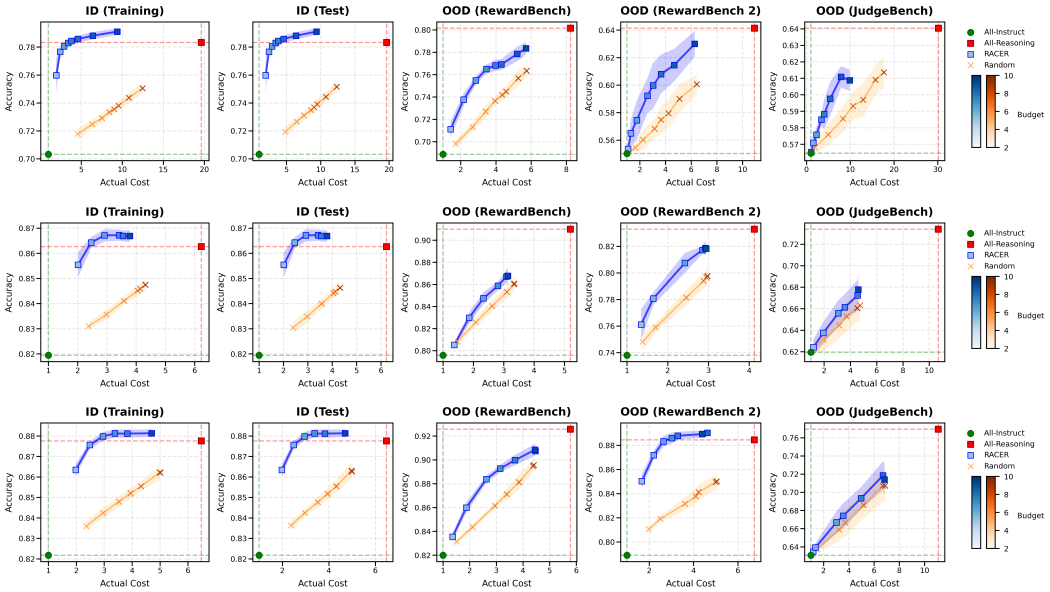


Figure 4: Routing performance on ID and OOD benchmarks (top to bottom: Qwen3-1.7B/4B/8B).

OffsetBias as the OOD test set. Judgments are produced by the hybrid reasoning model Qwen3-4B (Yang et al., 2025). Reward is defined as correctness, and cost as the relative token usage between reasoning and non-reasoning judgments (Table C.1). We use bge-3 embeddings (Chen et al., 2024a) to get embeddings of the query and responses, and a linear router.

As shown in the left panel of Figure 3, the OOD shift favors lower-cost queries, all methods satisfy the budget constraint across runs. RACER-R consistently achieves the highest reward, with stronger robustness (lower temperature) yielding further gains, indicating that when OOD cost is low, emphasizing reward robustness enables more effective budget utilization. In contrast, in the right panel, RACER-C attains a lower reward than RACER-R and ACER but consistently maintains OOD costs within budget, demonstrating that cost robustness provides a safer strategy when in-distribution cost is low but OOD shifts increase violation risk. Overall, RACER variants exhibit greater robustness in both reward and cost compared to the non-robust baseline ACER.

5.3 Scaling Evaluation to Standard Benchmarks

Training data. Following the shared data generation protocol, we construct the training set from: (i) 20,000 instances from Skywork-Reward Liu et al. (2024a), and (ii) an additional 20,000 instances sampled from Math-Step-DPO-10K Lai et al. (2024) and Code-Preference-Pairs Vezora (2024). We include the math and code data to improve coverage of reasoning-intensive domains that are under-represented in Skywork-Reward. We generate judgments by pairing the reasoning and non-reasoning variants of Qwen3-1.7B/4B/8B.

Evaluation datasets. We evaluate the learned routing policy on standard LLM-as-a-Judge benchmarks spanning diverse task domains, including RewardBench Lambert et al. (2025), RewardBench-2 Malik et al. (2025), and JudgeBench Tan et al. (2024b). All evaluation benchmarks are held out from training and are used exclusively for out-of-distribution evaluation. Judgments are generated using the same model pairs.

Baselines. We compare against three baselines:
- **All-Instruct**, which always selects the instruct mode (cost fixed to 1 and accuracy equal to the instruct judge's average correctness);
- **All-Reasoning**, which always selects the reasoning mode (accuracy and cost given by the reasoning judge's average correctness and token consumption, respectively);
- **Random**, which activates the reasoning mode independently for each instance with probability equal to the learned policy's average reasoning rate.

We evaluate the routing policy under compute budgets $B \in \{2, 2.5, 3, 3.5, 4, 5, 7, 10\}$. For each budget, we repeat training and evaluation 20 times and report the mean and standard deviation of accuracy and realized cost ratio. Our routing policy is parametrized by a four-layer neural network. See Appendix C.3 and C.5 for detailed implementation and hyperparameter choices.

Fig. 4 reports accuracy–cost trade-offs under varying budgets for three model scales. On both the training split and the in-distribution test split, RACER achieves a notably favorable regime: at roughly half of the All-Reasoning cost, it matches and often surpasses the All-Reasoning judge's accuracy, indicating that reasoning can be concentrated on the subset of instances where it delivers the largest marginal benefit. Under OOD benchmarks, RACER continues to yield consistent improvements over the baselines, demonstrating that the learned selection rule generalizes beyond the training distribution. The Random baseline forms an approximately linear interpolation between the All-Instruct and All-Reasoning endpoints, corresponding to indiscriminate activation of reasoning at a fixed rate. In contrast, RACER traces a concave, higher-accuracy frontier; the area between the RACER curve and the Random curve reflects the additional gain from strategic instance-level selection, which cannot be explained by merely increasing the overall reasoning rate, but instead by allocating reasoning to the right examples under the same budget.

6 Conclusion and Limitation

In this work, we study whether reasoning capabilities acquired by LLMs from problem-solving tasks transfer to judgment tasks. We find that reasoning-based judges can substantially improve accuracy, but the gains are task-dependent and often incur significantly higher computational cost, particularly under distribution shift. To address these problems, we propose RACER, which adaptively activates reasoning judges under a fixed budget. Empirically, RACER consistently outperforms baselines on multiple benchmarks, achieving superior accuracy–cost trade-offs. We are also the first to provide a theoretical analysis with linear convergence guarantees, supporting both efficiency and robustness.

REFERENCES

Pranjal Aggarwal, Aman Madaan, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, et al. Automix: Automatically mixing language models. *arXiv preprint arXiv:2310.12963*, 2023.

Alexandre Belloni, Victor Chernozhukov, and Lie Wang. Square-root lasso: pivotal recovery of sparse signals via conic programming. *Biometrika*, 98(4):791–806, 2011.

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, et al. Llama-nemotron: Efficient reasoning models. *arXiv preprint arXiv:2505.00949*, 2025.

Shicong Cen, Chen Cheng, Yuxin Chen, Yuting Wei, and Yuejie Chi. Fast global convergence of natural policy gradient methods with entropy regularization. *Operations Research*, 70(4): 2563–2578, 2022.

Souradip Chakraborty, Jiahao Qiu, Hui Yuan, Alec Koppel, Furong Huang, Dinesh Manocha, Amrit Bedi, and Mengdi Wang. Maxmin-rlhf: Towards equitable alignment of large language models with diverse human preferences. In *ICML 2024 Workshop on Models of Human Feedback for AI Alignment*, 2024.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*, 4(5), 2024a.

Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

Nuo Chen, Zhiyuan Hu, Qingyun Zou, Jiaying Wu, Qian Wang, Bryan Hooi, and Bingsheng He. Judgelrm: Large reasoning models as a judge. *arXiv preprint arXiv:2504.00050*, 2025a.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*, 2024b.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, et al. Rm-r1: Reward modeling as reasoning. *arXiv preprint arXiv:2505.02387*, 2025b.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Distributionally robust federated averaging. *Advances in neural information processing systems*, 33:15111–15122, 2020.

Dongsheng Ding, Kaiqing Zhang, Jiali Duan, Tamer Başar, and Mihailo R Jovanović. Convergence and sample complexity of natural policy gradient primal-dual methods for constrained mdps. *arXiv preprint arXiv:2206.02346*, 2022.

Dongsheng Ding, Chen-Yu Wei, Kaiqing Zhang, and Alejandro Ribeiro. Last-iterate convergent policy gradient primal-dual methods for constrained mdps. *Advances in Neural Information Processing Systems*, 36:66138–66200, 2023.

John C Duchi and Hongseok Namkoong. Learning models with uniform performance via distributionally robust optimization. *The Annals of Statistics*, 49(3):1378–1406, 2021.

John C Duchi, Peter W Glynn, and Hongseok Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *Mathematics of Operations Research*, 46(3):946–969, 2021.

Evan Frick, Connor Chen, Joseph Tennyson, Tianle Li, Wei-Lin Chiang, Anastasios N Angelopoulos, and Ion Stoica. Prompt-to-leaderboard. *arXiv preprint arXiv:2502.14855*, 2025.

Jinlan Fu, See Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6556–6576, 2024.

Uri Gadot, Kaixin Wang, Navdeep Kumar, Kfir Yehuda Levy, and Shie Mannor. Bring your own (non-robust) algorithm to solve robust mdps by estimating the worst kernel. In *Forty-first International Conference on Machine Learning*, 2024.

Joel Goh and Melvyn Sim. Distributionally robust optimization and its tractable approximations. *Operations research*, 58(4-part-1):902–917, 2010.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Zhaolin Hu and L Jeff Hong. Kullback-leibler divergence constrained distributionally robust optimization. *Available at Optimization Online*, 1(2):9, 2013.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, et al. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*, 2023.

Daniel Kuhn, Peyman Mohajerin Esfahani, Viet Anh Nguyen, and Soroosh Shafieezadeh-Abadeh. Wasserstein distributionally robust optimization: Theory and applications in machine learning. In *Operations research & management science in the age of analytics*, pp. 130–166. Informs, 2019.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*, 2024.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, Lester James Validad 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.

Daniel Levy, Yair Carmon, John C Duchi, and Aaron Sidford. Large-scale methods for distributionally robust optimization. *Advances in neural information processing systems*, 33:8847–8860, 2020.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`, 5 2023.

Guosheng Liang, Longguang Zhong, Ziyi Yang, and Xiaojun Quan. Thinkswitcher: When to think hard, when to think fast. *arXiv preprint arXiv:2505.14183*, 2025.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*, 2024a.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*, 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. *arXiv preprint arXiv:2410.16184*, 2024b.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A. Smith, Hannaneh Hajishirzi, and Nathan Lambert. Rewardbench 2: Advancing reward model evaluation, 2025. URL `https://arxiv.org/abs/2506.01937`.

Mirco Mutti, Riccardo De Santi, Piersilvio De Bartolomeis, and Marcello Restelli. Convex reinforcement learning in finite trials. *Journal of Machine Learning Research*, 24(250):1–42, 2023.

Hongseok Namkoong and John C Duchi. Stochastic gradient methods for distributionally robust optimization with f-divergences. *Advances in neural information processing systems*, 29, 2016.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025.

OpenAI. Learning to reason with LLMs. `https://openai.com/index/learning-to-reason-with-llms/`, 2024. Accessed: 2025-12-28.

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.

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.

Hamed Rahimian and Sanjay Mehrotra. Distributionally robust optimization: A review. *arXiv preprint arXiv:1908.05659*, 2019.

Halsey Lawrence Royden and Patrick Fitzpatrick. *Real analysis*, volume 32. Macmillan New York, 1988.

Swarnadeep Saha, Xian Li, Marjan Ghazvininejad, Jason Weston, and Tianlu Wang. Learning to plan & reason for evaluation with thinking-llm-as-a-judge. *arXiv preprint arXiv:2501.18099*, 2025.

Soroosh Shafieezadeh Abadeh, Peyman M Mohajerin Esfahani, and Daniel Kuhn. Distributionally robust logistic regression. *Advances in neural information processing systems*, 28, 2015.

Maurice Sion. On general minimax theorems. *Pacific Journal of Mathematics*, 8:171–176, 1958.

Seongho Son, William Bankes, Sayak Ray Chowdhury, Brooks Paige, and Ilija Bogunovic. Right now, wrong then: Non-stationary direct preference optimization under preference drift. *arXiv preprint arXiv:2407.18676*, 2024.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. Judgebench: A benchmark for evaluating llm-based judges. *arXiv preprint arXiv:2410.12784*, 2024a.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, Willian Yuan Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. Judgebench: A benchmark for evaluating llm-based judges, 2024b. URL `https://arxiv.org/abs/2410.12784`.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Vezora. Code-preference-pairs dpo mix. `https://huggingface.co/datasets/Vezora/Code-Preference-Pairs`, 2024. Hugging Face dataset.

Chen-Yu Wei, Chung-Wei Lee, Mengxiao Zhang, and Haipeng Luo. Linear last-iterate convergence in constrained saddle-point optimization. In *International Conference on Learning Representations*, 2021.

Chenxi Whitehouse, Tianlu Wang, Ping Yu, Xian Li, Jason Weston, Ilia Kulikov, and Swarnadeep Saha. J1: Incentivizing thinking in llm-as-a-judge via reinforcement learning. *arXiv preprint arXiv:2505.10320*, 2025.

Zaiyan Xu, Sushil Vemuri, Kishan Panaganti, Dileep Kalathil, Rahul Jain, and Deepak Ramachandran. Robust llm alignment via distributionally robust direct preference optimization. *arXiv preprint arXiv:2502.01930*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Wencheng Zhang, Shiqin Qiao, Lingjie Luo, Yinfeng Li, Chuanyang Zheng, Qian Xu, Meng Li, Yong Gui, Yijun He, Jianing Qiu, et al. Synapseroute: An auto-route switching framework on dual-state large language model. *arXiv preprint arXiv:2507.02822*, 2025.

Zhehao Zhang, Ryan A Rossi, Branislav Kveton, Yijia Shao, Diyi Yang, Hamed Zamani, Franck Dernoncourt, Joe Barrow, Tong Yu, Sungchul Kim, et al. Personalization of large language models: A survey. *arXiv preprint arXiv:2411.00027*, 2024.

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.

APPENDIX

The appendix provides additional theoretical analysis, experimental details, and supplementary experimental results. In Section B, we establish the existence and uniqueness of the saddle point and prove last-iterate convergence, providing theoretical guarantees for the convergence and optimality of the proposed algorithm, RACER. In Section C, we describe in detail the data generation process, the training procedure, and the evaluation protocol. In Section D, we report supplementary results, including training curves and representative case studies that further motivate our proposed algorithm.

A RELATED WORK

LLM-as-a-Judge and Reasoning. Human evaluation is considered the gold-standard metric for assessing LLM-generated content (Ouyang et al., 2022; Zheng et al., 2023). However, it is costly and time-consuming, making it difficult to scale in practice. To address this, LLM-as-a-Judge has been proposed as an automatic proxy (Zheng et al., 2023; Li et al., 2023; Liu et al., 2023; Kim et al., 2023; Fu et al., 2024). One advantage is that the LLM can provide explanations for its final judgments, which facilitates error analysis. To evaluate how well LLM-as-a-Judge truly performs, benchmarks have been introduced to measure the accuracy of judgments across different domains (Tan et al., 2024a; Liu et al., 2024b; Lambert et al., 2025). There are several recent studies have shown that incorporating reasoning into models through reinforcement learning on judge-specific tasks can further enhance evaluation performance (Whitehouse et al., 2025; Saha et al., 2025; Chen et al., 2025a;b). Notably, our work demonstrates that even without judge-specific training, reasoning abilities acquired from general-domain training can still effectively transfer.

LLM Routing. As the number of parameters in LLMs continues to grow, their inference cost increases substantially. To address this issue, LLM routing has emerged as an effective strategy that assigns user queries to appropriate LLMs while balancing both performance and cost. Chen et al. (2023) adopted a cascading strategy, sequentially querying LLMs until a satisfactory response is produced. P2L (Frick et al., 2025) employs a prompt-to-regression approach to predict a vector of Bradley–Terry coefficients, which are then used to select the optimal model. Aggarwal et al. (2023) and Ong et al. (2025) trained a binary predictor to switch between a strong and weak model. While effective, this approach requires deploying multiple LLMs simultaneously. Recent studies (Liang et al., 2025; Zhang et al., 2025) investigate mode switching within a single hybrid reasoning model, which is most closely related to our setting. However, we introduce a principled policy learning framework that not only maximizes performance under budget constraints, but also explicitly addresses **distribution shift**, an important yet largely overlooked challenge in existing routing systems. Furthermore, all prior studies predominantly focus on question-answering tasks, whereas we present the first comprehensive investigation of routing strategies in the LLM-as-judge domain.

Distributionally Robust Learning. Distributionally Robust Optimization (DRO) has been extensively studied in machine learning (Shafieezadeh Abadeh et al., 2015; Deng et al., 2020), statistics (Belloni et al., 2011; Duchi & Namkoong, 2021), and operations research (Goh & Sim, 2010; Kuhn et al., 2019). It is typically formulated as a minimax problem, in which an adversary perturbs the data-generating distribution within a prescribed uncertainty set to maximize the expected loss, while the learner optimizes model parameters to minimize this worst-case risk. Among various constructions of uncertainty sets, f-divergence balls are a commonly used choice (Hu & Hong, 2013; Namkoong & Duchi, 2016; Levy et al., 2020; Duchi & Namkoong, 2021), due to their strong connections with classical divergence measures and well-studied statistical properties that facilitate analysis and implementation in DRO frameworks.

B THEORETICAL ANALYSIS

B.1 DERIVING DATA DISTRIBUTION FROM KL UNCERTAINTY SET

Lemma B.1. *Let $\rho_n \in \Delta_n$ be the empirical distribution over $\{z_i\}_{i=1}^n$, and define $f_i := \mathbb{E}_{a\sim\pi(\cdot|z_i)}[f(z_i, a)]$. Let*

$$\mathcal{U}(\rho_n, \delta) := \Big\{\tilde{\rho} \in \Delta_n : \sum_{i=1}^{n} \tilde{\rho}(i) \log \frac{\tilde{\rho}(i)}{\rho_n(i)} \le \delta\Big\}.$$

Define $\underline{\rho}$ and $\overline{\rho}$ as the solutions to

$$\underline{\rho} \in \arg\min_{\tilde{\rho} \in \mathcal{U}(\rho_n, \delta)} \sum_{i=1}^{n} \tilde{\rho}(i) f_i, \qquad \overline{\rho} \in \arg\max_{\tilde{\rho} \in \mathcal{U}(\rho_n, \delta)} \sum_{i=1}^{n} \tilde{\rho}(i) f_i.$$

Then there exist $\underline{s}, \overline{s} \in \mathbb{R}$ and $\tau > 0$ such that, for all i,

$$\underline{\rho}(i) \propto \rho_n(i) \exp\left(\frac{\underline{s} - f_i}{\tau}\right), \qquad \overline{\rho}(i) \propto \rho_n(i) \exp\left(\frac{f_i - \overline{s}}{\tau}\right),$$

with $\underline{s} \leq \sum_{i=1}^{n} \rho_n(i) f_i$ and $\overline{s} \geq \sum_{i=1}^{n} \rho_n(i) f_i$.

Proof. We prove the claim for $\underline{\rho}$; the proof for $\overline{\rho}$ is analogous.

Consider the convex optimization problem

$$\begin{aligned} \underset{\tilde{\rho} \in \mathbb{R}^n}{\text{minimize}} \quad & \sum_{i=1}^{n} \tilde{\rho}(i) f_i \\ \text{subject to} \quad & \sum_{i=1}^{n} \tilde{\rho}(i) \log \frac{\tilde{\rho}(i)}{\rho_n(i)} \leq \delta, \\ & \sum_{i=1}^{n} \tilde{\rho}(i) = 1, \\ & \tilde{\rho}(i) \geq 0, \ \forall i. \end{aligned} \tag{B.1}$$

The objective is linear and the KL constraint is convex. Moreover, Slater's condition holds (e.g., $\tilde{\rho} = \rho_n$ is strictly feasible when $\delta > 0$). Hence, the Karush–Kuhn–Tucker (KKT) conditions are necessary and sufficient for optimality.

Introducing Lagrange multipliers $\lambda \geq 0$ for the KL constraint, $\mu \in \mathbb{R}$ for normalization, and $\nu_i \geq 0$ for non-negativity, the Lagrangian of (B.1) is

$$\mathcal{L}(\tilde{\rho}, \lambda, \mu, \nu) = \sum_{i=1}^{n} \tilde{\rho}(i) f_i + \lambda \left(\sum_{i=1}^{n} \tilde{\rho}(i) \log \frac{\tilde{\rho}(i)}{\rho_n(i)} - \delta \right) + \mu \left(\sum_{i=1}^{n} \tilde{\rho}(i) - 1 \right) - \sum_{i=1}^{n} \nu_i \tilde{\rho}(i). \tag{B.2}$$

The stationarity condition of the KKT implies that, for all i,

$$f_i + \lambda \left(\log \frac{\tilde{\rho}(i)}{\rho_n(i)} + 1 \right) + \mu - \nu_i = 0. \tag{B.3}$$

Solving (B.3) for $\tilde{\rho}(i)$ yields

$$\tilde{\rho}(i) = \rho_n(i) \ \exp\left(-\frac{f_i + \mu + \lambda - \nu_i}{\lambda} \right). \tag{B.4}$$

In the non-degenerate case (e.g., $\delta > 0$ and $\{f_i\}$ not all equal), the KL constraint is active at the optimum, and thus $\lambda > 0$. Since $\rho_n(i) > 0$ and the exponential is strictly positive, Eq. (B.4) implies $\tilde{\rho}(i) > 0$ for all i. By complementary slackness, we therefore have $\nu_i = 0$ for all i, and Eq. (B.4) simplifies to

$$\tilde{\rho}(i) = \rho_n(i) \ \exp\left(-\frac{f_i + \mu + \lambda}{\lambda} \right). \tag{B.5}$$

Let $\tau \coloneqq \lambda$ and define $\underline{s} \coloneqq -(\mu + \lambda)$. Then Eq. (B.5) can be written as

$$\tilde{\rho}(i) \propto \rho_n(i) \ \exp\left(\frac{\underline{s} - f_i}{\tau} \right), \tag{B.6}$$

which establishes the stated form of $\underline{\rho}$.

It remains to show that $\underline{s} \leq \sum_i \rho_n(i) f_i$. Using the normalization condition $\sum_i \tilde{\rho}(i) = 1$ and Eq. (B.6), we obtain

$$\exp\Big(\frac{\underline{s}}{\tau}\Big) = \frac{1}{\sum_{i=1}^n \rho_n(i)\, \exp\Big(-\frac{f_i}{\tau}\Big)}. \tag{B.7}$$

Applying Jensen's inequality to the convex function $\exp(\cdot)$ yields

$$\exp\left(-\frac{\sum_{i=1}^n \rho_n(i) f_i}{\tau}\right) \leq \sum_{i=1}^n \rho_n(i)\, \exp\left(-\frac{f_i}{\tau}\right). \tag{B.8}$$

Combining (B.7) and (B.8) gives

$$\exp\Big(\frac{\underline{s}}{\tau}\Big) \leq \exp\left(\frac{\sum_{i=1}^n \rho_n(i) f_i}{\tau}\right),$$

which implies $\underline{s} \leq \sum_{i=1}^n \rho_n(i) f_i$.

The proof for $\overline{\rho}$ follows identically by applying the same argument to $\max_{\tilde{\rho} \in \mathcal{U}(\rho_n, \delta)} \sum_i \tilde{\rho}(i) f_i$ (equivalently, minimizing $-\sum_i \tilde{\rho}(i) f_i$), yielding $\overline{\rho}(i) \propto \rho_n(i) \exp((f_i - \overline{s})/\tau)$ and $\overline{s} \geq \sum_i \rho_n(i) f_i$. □

B.2 Existence and Uniqueness of the Saddle Point (Theorem 4.1)

We first present Proposition B.2, which states the Slater condition (feasibility) and is commonly required in the duality analysis of constrained optimization (see, e.g., Ding et al., 2022; 2023).

B.2.1 Feasibility Check

Proposition B.2 (Feasibility)**.** *There exist a constant $\xi > 0$ and a policy $\pi_0 \in \Pi$ such that $\mathbb{E}_{(z,l)\sim\underline{\rho}, a\sim\pi_0(\cdot|z)}[c(z,a)] - C \leq -\xi$, where $\pi_0(0 \mid z) = 1$ and $\pi_0(1 \mid z) = 0$ for all $z \in \mathcal{Z}$.*

Proposition B.2 is straightforward to verify. Since $\mathbb{E}_{(z,l)\sim\underline{\rho}, a\sim\pi_0(\cdot|z)}[c(z,a)] = 1$, the inequality holds whenever $\xi \leq C - 1$ with $C > 1$. To analyze the saddle point of $\mathcal{L}_\beta(\pi, \lambda)$, we define π^* as the optimal solution to (8), that is,

$$\pi^* \in \arg\max_{\pi \in \Pi} \min_{\lambda \geq 0} \mathcal{L}_\beta(\pi, \lambda),$$

and let λ^* denote the corresponding optimal dual variable,

$$\lambda^* \in \arg\min_{\lambda \geq 0} \max_{\pi \in \Pi} \mathcal{L}_\beta(\pi, \lambda),$$

with Λ^* denoting the set of all optimal dual variables. We denote $\mathcal{L}_\beta^\lambda \coloneqq \max_{\pi \in \Pi} \mathcal{L}_\beta(\pi, \lambda)$. We can then show that Λ^* is bounded as established in Lemma B.3.

B.2.2 Boundness of Solution Space

Lemma B.3 (Boundness of λ^*)**.** *The optimal dual variable λ^* satisfies*

$$0 \leq \lambda^* \leq \frac{\mathcal{L}_\beta^{\lambda^*} - \mathbb{P}_{\underline{\rho}}(\phi_0(z) = l)}{\xi},$$

and hence Λ^* is bounded.

Proof. Let $\Lambda_a \coloneqq \{\lambda \geq 0 \mid \mathcal{L}_\beta^\lambda \leq a\}$ be a sublevel set of the dual objective for $a \in \mathbb{R}$. Recall that $\mathcal{L}_\beta^\lambda = \max_{\pi \in \Pi} \mathcal{L}_\beta(\pi, \lambda)$. For any $\lambda \in \Lambda_a$, we have

$$\begin{aligned}
a \geq \mathcal{L}_\beta^\lambda &\geq \mathcal{L}_\beta(\pi_0, \lambda) \\
&\geq \mathbb{E}_{(z,l)\sim\underline{\rho}, a\sim\pi_0(\cdot|z)}[r(z,a,l)] - \lambda \left(\mathbb{E}_{(z,l)\sim\overline{\rho}, a\sim\pi_0(\cdot|z)}[c(z,a)] - C\right) + \beta \left(\mathbb{E}_{(z,l)\sim\rho}\left[\mathcal{H}\left(\pi_0(\cdot \mid z)\right)\right] + \frac{1}{2}\lambda^2\right) \\
&\geq \mathbb{E}_{(z,l)\sim\underline{\rho}, a\sim\pi_0(\cdot|z)}[r(z,a,l)] + \lambda\xi + \beta \left(\mathbb{E}_{(z,l)\sim\rho}\left[\mathcal{H}\left(\pi_0(\cdot \mid z)\right)\right] + \frac{1}{2}\lambda^2\right),
\end{aligned}$$

where the last inequality comes from Proposition B.2. Because $\mathcal{H}\left(\pi_0(\cdot \mid z)\right) = 0$ and

$$\mathbb{E}_{(z,l)\sim\underline{\rho},a\sim\pi_0(\cdot|z)}[r(z,a,l)] = \mathbb{P}_{\underline{\rho}}(\phi_0(z) = l),$$

it follows that

$$a \geq \mathcal{L}_\beta^\lambda \geq \mathbb{P}_{\underline{\rho}}(\phi_0(z) = l) + \lambda\xi.$$

Thus we get $\lambda \leq \frac{\mathcal{L}_\beta^\lambda - \mathbb{P}_{\underline{\rho}}(\phi_0(z)=l)}{\xi}$. Finally, setting $a = \mathcal{L}_\beta^{\lambda^*}$ with $\Lambda_a = \Lambda^*$, we derive

$$0 \leq \lambda^* \leq \frac{\mathcal{L}_\beta^{\lambda^*} - \mathbb{P}_{\underline{\rho}}(\phi_0(z) = l)}{\xi}.$$

□

Based on Lemma B.3, we define the domain for λ as

$$\Lambda = [0, \frac{\mathcal{L}_\beta^{\lambda^*} - \mathbb{P}_{\underline{\rho}}(\phi_0(z) = l)}{\xi}],$$

which ensures $\Lambda^* \subseteq \Lambda$. With this setup, all the required conditions for applying Sion's minimax theorem (Sion, 1958) are satisfied, which ensures the existence and uniqueness of the regularized saddle point.

B.2.3 Proof of Theorem 4.1

Proof. Based on Assumption 1, Π is a convex set. Moreover, since Λ is convex and bounded, i.e., compact, Sion's minimax theorem (Sion, 1958) guarantees the existence of a saddle point of $\mathcal{L}_\beta(\pi, \lambda)$ in $\Pi \times \Lambda$. We next prove the uniqueness of this saddle point. Recall that

$$\mathcal{L}_\beta(\pi,\lambda) = \mathbb{E}_{(z,l)\sim\underline{\rho},a\sim\pi(\cdot|z)}[r(z,a,l)] - \lambda\left(\mathbb{E}_{(z,l)\sim\overline{\rho},a\sim\pi(\cdot|z)}[c(z,a)] - C\right) + \beta\left(\mathbb{E}_{(z,l)\sim\rho}\left[\mathcal{H}\left(\pi(\cdot \mid z)\right)\right] + \frac{1}{2}\lambda^2\right).$$

Since $\frac{\partial^2 \mathcal{L}_\beta(\pi,\lambda)}{\partial\lambda^2} = \beta > 0$, $\mathcal{L}_\beta(\pi, \lambda)$ is strictly convex in λ. Then we will show $\mathcal{L}_\beta(\pi, \lambda)$ is strictly concave in π. We first examine the expectation terms:

$$\mathbb{E}_{(z,l)\sim\underline{\rho},a\sim\pi(\cdot|z)}[r(z,a,l)] = \mathbb{E}_{(z,l)\sim\underline{\rho}}\left(\sum_{a=0}^{1} r(z,a,l)\pi(a|z)\right),$$

and

$$\mathbb{E}_{(z,l)\sim\overline{\rho},a\sim\pi(\cdot|z)}c(z,a) = \mathbb{E}_{(z,l)\sim\overline{\rho}}\left(\sum_{a=0}^{1} c(z,a)\pi(a|z)\right).$$

Thus, both $\mathbb{E}_{(z,l)\sim\underline{\rho},a\sim\pi(\cdot|z)}[r(z,a,l)]$ and $\mathbb{E}_{(z,l)\sim\overline{\rho},a\sim\pi(\cdot|z)}c(z,a)$ are linear in π. Because the entropy term $\mathcal{H}\left(\pi(\cdot \mid z)\right)$ is strongly concave in π, it follows that its expectation $\mathbb{E}_{(z,l)\sim\rho}\left[\mathcal{H}\left(\pi(\cdot \mid z)\right)\right]$ remains strongly concave in π. Consequently, we have $\mathcal{L}_\beta(\pi, \lambda)$ is also strongly concave in π for any fixed λ.

Now we have shown $\mathcal{L}_\beta(\pi, \lambda)$ is strictly concave in π and strictly convex in λ, which implies saddle point (π^*, λ^*) is unique.

The existence of a saddle point as guaranteed by Lemma B.3, further implies that for any $(\pi, \lambda) \in \Pi \times \Lambda$,

$$\mathcal{L}_\beta(\pi, \lambda^*) \leq \mathcal{L}_\beta(\pi^*, \lambda^*) \leq \mathcal{L}_\beta(\pi^*, \lambda).$$

The first inequality indicates that for any $\pi \in \Pi$,

$$\begin{aligned}\mathcal{L}(\pi^*,\lambda^*) + \beta\left(\mathbb{E}_{(z,l)\sim\rho}\left[\mathcal{H}\left(\pi^*(\cdot \mid z)\right)\right] + \frac{1}{2}(\lambda^*)^2\right) &\geq \mathcal{L}(\pi,\lambda^*) + \beta\left(\mathbb{E}_{(z,l)\sim\rho}\left[\mathcal{H}\left(\pi(\cdot \mid z)\right)\right] + \frac{1}{2}(\lambda^*)^2\right) \\ &\geq \mathcal{L}(\pi,\lambda^*) + \frac{\beta}{2}(\lambda^*)^2,\end{aligned}$$

while the second inequality implies that for any $\lambda \in \Lambda$,

$$\mathcal{L}(\pi^*, \lambda) + \beta \left(\mathbb{E}_{(z,l)\sim\rho} \left[\mathcal{H} \left(\pi^*(\cdot \mid z) \right) \right] + \frac{1}{2}\lambda^2 \right) \geq \mathcal{L}(\pi^*, \lambda^*) + \beta \left(\mathbb{E}_{(z,l)\sim\rho} \left[\mathcal{H} \left(\pi^*(\cdot \mid z) \right) \right] + \frac{1}{2}(\lambda^*)^2 \right)$$
$$\geq \mathcal{L}(\pi^*, \lambda^*) + \beta\, \mathbb{E}_{(z,l)\sim\rho} \left[\mathcal{H} \left(\pi^*(\cdot \mid z) \right) \right].$$

Combining the two inequalities above shows that (π^*, λ^*) is a saddle point of the original Lagrangian $\mathcal{L}(\pi, \lambda)$, up to two β-regularization terms. □

B.3 Proof of Last-iterate Convergence (Theorem 4.2)

In practice, we will update (π_t, λ_t) as the following steps.

$$\pi_{t+1} = \arg\max_{\pi\in\Pi} \mathbb{E}_{z\sim\underline{\rho}, a\sim\pi(\cdot|z)} \left[r(z,a) \right] - \lambda_t \mathbb{E}_{z\sim\overline{\rho}, a\sim\pi(\cdot|z)} [g(z,a)] + \beta\, \mathbb{E}_{z\sim\rho} \left[\mathcal{H} \left(\pi(\cdot \mid z) \right) \right], \tag{B.9}$$

$$\lambda_{t+1} = \left[\lambda_t + \eta \left(\mathbb{E}_{z\sim\overline{\rho}, a\sim\pi_{t+1}(\cdot|z)} [g(z,a)] - C - \beta\lambda_t \right) \right]_+, \tag{B.10}$$

where η denotes the step size.

For notation simplicity, we denote $w_1(z) = p_{\underline{\rho}}(z)/p_\rho(z)$ and $w_2(z) = p_{\overline{\rho}}(z)/p_\rho(z)$ for any $z \in \mathcal{Z}$, which are well defined under Assumption 3.

Step 1. We first derive the closed-form solution to (6).

According to (6), we optimize the following objective

$$\max_{\pi\in\Pi} \mathbb{E}_{(z,l)\sim\underline{\rho}, a\sim\pi(\cdot|z)} [r(z,a,l)] - \lambda_t \mathbb{E}_{(z,l)\sim\overline{\rho}, a\sim\pi(\cdot|z)} [c(z,a)] + \beta\, \mathbb{E}_{(z,l)\sim\rho} \left[\mathcal{H} \left(\pi(\cdot \mid z) \right) \right]. \tag{B.11}$$

Analogously to arguments used in the proof of DPO (Rafailov et al., 2023), (B.11) can be rewritten as

$$\begin{aligned}
&\max_{\pi\in\Pi} \mathbb{E}_{(z,l)\sim\underline{\rho}, a\sim\pi(\cdot|z)} [r(z,a,l)] - \lambda_t \mathbb{E}_{(z,l)\sim\overline{\rho}, a\sim\pi(\cdot|z)} [c(z,a)] + \beta\, \mathbb{E}_{(z,l)\sim\rho} \left[\mathcal{H} \left(\pi(\cdot \mid z) \right) \right] \qquad \text{(B.12)} \\
=&\max_{\pi\in\Pi} \mathbb{E}_{(z,l)\sim\rho, a\sim\pi(\cdot|z)} \left[w_1(z) r(z,a,l) - \lambda_t w_2(z) c(z,a) \right] + \beta\, \mathbb{E}_{(z,l)\sim\rho} \left[\mathcal{H} \left(\pi(\cdot \mid z) \right) \right] \\
=&\max_{\pi\in\Pi} \mathbb{E}_{(z,l)\sim\rho, a\sim\pi(\cdot|z)} \left[w_1(z) r(z,a,l) - \lambda_t w_2(z) c(z,a) \right] - \beta\, \mathbb{E}_{(z,l)\sim\rho, a\sim\pi(\cdot|z)} \left[\log \left(\pi(a|z) \right) \right] \\
=&\beta \max_{\pi\in\Pi} \mathbb{E}_{(z,l)\sim\rho} \mathbb{E}_{a\sim\pi(\cdot|z)} \left[\frac{1}{\beta} w_1(z) r(z,a,l) - \frac{\lambda_t}{\beta} w_2(z) c(z,a) - \log \left(\pi(a|z) \right) \right] \\
=&\beta \max_{\pi\in\Pi} \mathbb{E}_{(z,l)\sim\rho} \mathbb{E}_{a\sim\pi(\cdot|z)} \left[\log \frac{\exp\left(\frac{1}{\beta} w_1(z) r(z,a,l) \right)}{\exp\left(\frac{\lambda_t}{\beta} w_2(z) c(z,a) \right) \pi(a|z)} \right] \\
=&-\beta \min_{\pi\in\Pi} \mathbb{E}_{(z,l)\sim\rho} \mathbb{E}_{a\sim\pi(\cdot|z)} \left[\log \frac{\pi(a|z)}{\exp\left(\frac{1}{\beta} w_1(z) r(z,a,l) - \frac{\lambda_t}{\beta} w_2(z) c(z,a) \right)} \right] \\
=&-\beta \min_{\pi\in\Pi} \mathbb{E}_{(z,l)\sim\rho} \mathbb{E}_{a\sim\pi(\cdot|z)} \left[\log \frac{\pi(a|z)}{\exp\left(\frac{1}{\beta} w_1(z) r(z,a,l) - \frac{\lambda_t}{\beta} w_2(z) c(z,a) \right) \frac{1}{Q(z)}} - \log(Q(z)) \right],
\end{aligned} \tag{B.13}$$

where we have partition function:

$$Q(z, \lambda_t) = \sum_{a=0}^{1} \exp\left(\frac{1}{\beta} w_1(z) r(z,a,l) - \frac{\lambda_t}{\beta} w_2(z) c(z,a) \right).$$

Note that the partition function depends only on z, λ_t and is independent of the policy π. We now define

$$\pi_{t+1}(a|z) = \frac{1}{Q(z, \lambda_t)} \exp\left(\frac{1}{\beta} w_1(z) r(z,a,l) - \frac{\lambda_t}{\beta} w_2(z) c(z,a) \right),$$

which forms a valid probability distribution. Substituting this definition into (B.13), we can rewrite the objective as

$$-\beta \min_{\pi \in \Pi} \mathbb{E}_{(z,l)\sim\rho} \left[\mathrm{KL}(\pi(\cdot \mid z) \| \pi_{t+1}(\cdot \mid z)) - \log(Q(z, \lambda_t)) \right].$$

Since $Q(z, \lambda_t)$ does not depend on π, the minimum is achieved by the policy that minimizes the KL term. By Gibbs' inequality, the KL divergence attains its minimum value of zero if and only if the two distributions are identical. Therefore, the updated policy is given by

$$\pi_{t+1}(a|z) = \frac{1}{Q(z, \lambda_t)} \exp\left(\frac{1}{\beta} w_1(z) r(z, a, l) - \frac{\lambda_t}{\beta} w_2(z) c(z, a) \right). \tag{B.14}$$

Step 2. We next establish the convergence of λ_t.

We view π_{t+1} as a function of λ_t, and denote it by $\widetilde{\pi}_{\lambda_t} := \pi_{t+1}$. For notational convenience, we denote $d(\lambda) := \mathcal{L}_\beta^\lambda = \max_{\pi \in \Pi} \mathcal{L}_\beta(\pi, \lambda) = \mathcal{L}_\beta(\widetilde{\pi}_\lambda, \lambda)$. Then $d(\lambda)$ can be explicitly written as,

$$\begin{aligned} d(\lambda) =& \beta \, \mathbb{E}_{(z,l)\sim\rho} \left[\log(Q(z, \lambda)) \right] + \lambda C + \frac{\beta}{2} \lambda^2 \\ =& \beta \, \mathbb{E}_{(z,l)\sim\rho} \left[\log \sum_{a=0}^{1} \exp\left(\frac{1}{\beta} w_1(z) r(z, a, l) - \frac{\lambda}{\beta} w_2(z) c(z, a) \right) \right] + \lambda C + \frac{\beta}{2} \lambda^2. \end{aligned}$$

Consequently, the update rule in (7) can be expressed as $\lambda_{t+1} = [\lambda_t - \eta \, d'(\lambda_t)]_+$. We then compute the first and second derivatives of $d(\lambda)$. Notice that

$$d'(\lambda) = \beta \, \left(\mathbb{E}_{(z,l)\sim\rho} \left[\log(Q(z, \lambda)) \right] \right)' + C + \beta \lambda, \tag{B.15}$$

so it suffices to compute $\left(\mathbb{E}_{(z,l)\sim\rho} \left[\log(Q(z, \lambda)) \right] \right)'$. We have

$$\frac{\partial Q(z, \lambda)}{\partial \lambda} = \sum_{a=0}^{1} \exp\left(\frac{1}{\beta} w_1(z) r(z, a, l) - \frac{\lambda}{\beta} w_2(z) c(z, a) \right) \left(-\frac{1}{\beta} w_2(z) c(z, a) \right),$$

and therefore,

$$\begin{aligned} \frac{\partial \log Q(z, \lambda)}{\partial \lambda} &= \frac{\sum_{a=0}^{1} \exp\left(\frac{1}{\beta} w_1(z) r(z, a, l) - \frac{\lambda}{\beta} w_2(z) c(z, a) \right) \left(-\frac{1}{\beta} w_2(z) c(z, a) \right)}{\sum_{a=0}^{1} \exp\left(\frac{1}{\beta} w_1(z) r(z, a, l) - \frac{\lambda}{\beta} w_2(z) c(z, a) \right)} \\ &= \sum_{a=0}^{1} \widetilde{\pi}_\lambda(a|z) \left[-\frac{1}{\beta} w_2(z) c(z, a) \right] = \mathbb{E}_{a\sim\widetilde{\pi}_\lambda(\cdot|z)} \left[-\frac{1}{\beta} w_2(z) c(z, a) \right]. \end{aligned}$$

Based on Assumption 2 and 3, we have $| -\frac{\lambda}{\beta} w_2(z) c(z, a)| \leq \frac{MK}{\beta}$, thus

$$\left| \frac{\partial \log Q(z, \lambda)}{\partial \lambda} \right| \leq \frac{MK}{\beta},$$

which ensures the boundedness of the derivative. Hence, by Royden & Fitzpatrick (1988), the order of differentiation and expectation can be interchanged. Thus, (B.15) comes to be

$$d'(\lambda) = -\mathbb{E}_{(z,l)\sim\rho, a\sim\widetilde{\pi}_\lambda(\cdot|z)} w_2(z) c(z, a) + C + \beta \lambda.$$

For the second derivative, we have

$$d''(\lambda) = -\left(\mathbb{E}_{(z,l)\sim\rho, a\sim\widetilde{\pi}_\lambda(\cdot|z)} w_2(z) c(z, a) \right)' + \beta, \tag{B.16}$$

where

$$\left(\mathbb{E}_{a\sim\widetilde{\pi}_\lambda(\cdot|z)} w_2(z) c(z, a) \right)' = w_2(z) \sum_{a=0}^{1} \frac{\partial \widetilde{\pi}_\lambda(a|z)}{\partial \lambda} c(z, a). \tag{B.17}$$

Let $s_\lambda(a) = \exp\left(\frac{1}{\beta} w_1(z) r(z, a, l) - \frac{\lambda}{\beta} w_2(z) c(z, a) \right)$, then $\widetilde{\pi}_\lambda(a|z) = s_\lambda(a)/((s_\lambda(0) + s_\lambda(1))$. By the chain rule, we have

$$\frac{\partial \widetilde{\pi}_\lambda(a|z)}{\partial \lambda} = \sum_{b=0}^{1} \frac{\partial \widetilde{\pi}_\lambda(a|z)}{\partial s_\lambda(b)} \frac{\partial s_\lambda(b)}{\partial \lambda}. \tag{B.18}$$

We can derive that

$$\frac{\partial s_\lambda(b)}{\partial \lambda} = s_\lambda(b)\left(-\frac{1}{\beta}w_2(z)c(z,b)\right), \tag{B.19}$$

and

$$\frac{\partial \widetilde{\pi}_\lambda(a|z)}{\partial s_\lambda(b)} = \frac{\mathbb{I}(a=b)s_\lambda(1-a) - \mathbb{I}(a\neq b)s_\lambda(a)}{\left(s_\lambda(0)+s_\lambda(1)\right)^2} = \frac{\mathbb{I}(a=b)\widetilde{\pi}_\lambda(1-a|z) - \mathbb{I}(a\neq b)\widetilde{\pi}_\lambda(a|z)}{s_\lambda(0)+s_\lambda(1)}. \tag{B.20}$$

Combining (B.19) and (B.20), (B.18) comes to be

$$\begin{aligned}\frac{\partial \widetilde{\pi}_\lambda(a|z)}{\partial \lambda} &= \widetilde{\pi}_\lambda(a|z)\sum_{b=0}^{1}\left(\delta_{ab} - \widetilde{\pi}_\lambda(b|z)\right)\left(-\frac{1}{\beta}w_2(z)c(z,b)\right) \\ &= \frac{w_2(z)}{\beta}\,\widetilde{\pi}_\lambda(a|z)\left[\left(\mathbb{E}_{a'\sim\widetilde{\pi}_\lambda(\cdot|z)}c(z,a')\right) - c(z,a)\right].\end{aligned} \tag{B.21}$$

Similarly, since (B.21) is bounded according to Assumptions 2 and 3, we can interchange the order of differentiation and expectation. Combining (B.17) and (B.21), (B.16) turns to be

$$\begin{aligned}d''(\lambda) &= \beta - \mathbb{E}_{(z,l)\sim\rho}\frac{1}{\beta}w_2(z)^2\sum_{a=0}^{1}\widetilde{\pi}_\lambda(a|z)\left[\left(\mathbb{E}_{a'\sim\widetilde{\pi}_\lambda(\cdot|z)}c(z,a')\right) - c(z,a)\right]c(z,a) \\ &= \beta + \frac{1}{\beta}\,\mathbb{E}_{(z,l)\sim\rho}w_2(z)^2\,\mathrm{Var}_{a\sim\widetilde{\pi}_\lambda(\cdot|z)}c(z,a).\end{aligned} \tag{B.22}$$

Again, due to Assumption 2 and 3, we have $c(z,a)\le M$ and $w_2(z)\le K$. Hence,

$$\beta \le d''(\lambda) \le \beta + \frac{M^2K^2}{\beta}.$$

We therefore conclude that $d(\lambda)$ is strongly convex with parameter β, and its gradient is Lipschitz continuous with constant $\beta + \frac{M^2K^2}{\beta}$. Then, by Theorem 2.1.5 in Nesterov (2013), choosing the step size as $\eta = 2/(\beta+\beta+M^2K^2/\beta) = 2\beta/(M^2K^2+2\beta^2)$ in (7), we have

$$|\lambda_{t+1}-\lambda^*| \le \left(\frac{\beta+\frac{M^2K^2}{\beta}-\beta}{\beta+\frac{M^2K^2}{\beta}+\beta}\right)^t|\lambda_0-\lambda^*| = \left(\frac{M^2K^2}{M^2K^2+2\beta^2}\right)^t|\lambda_0-\lambda^*|. \tag{B.23}$$

In our setting, although the update involves a projection onto the feasible set, the projection operator is non-expansive. Consequently, standard results for projected gradient descent apply, and the projected updates achieve the same convergence rate as the corresponding unconstrained gradient descent.

Step 3. Finally, we show that π_t converges to π^* using the KL divergence.

Given z, the quantity $\mathrm{KL}(\pi_t(\cdot|z)\|\pi^*(\cdot|z))$ is a Bregman divergence between the corresponding natural parameters of the exponential family:

$$D_\psi(\theta_t,\theta^*) = \mathrm{KL}(\pi_t(\cdot|z)\|\pi^*(\cdot|z)),$$

where the convex potential ψ is the log-partition function

$$\psi(\theta) = \log Q(z,-\theta\beta) = \log\sum_{a=0}^{1}\exp\left(\frac{1}{\beta}w_1(z)r(z,a,l) - \theta w_2(z)c(z,a)\right).$$

Here we let $\theta_t = -\lambda_t/\beta$ and $\theta^* = -\lambda^*/\beta$, treating $\widetilde{\pi}_\lambda(\cdot|z)$ as an exponential-family distribution with natural parameter θ.

Then we have

$$\psi'(\theta) = \frac{\exp\left(\frac{1}{\beta}w_1(z)r(z,a,l) - \theta w_2(z)c(z,a)\right)c(z,a)}{\sum_{a'=0}^{1}\exp\left(\frac{1}{\beta}w_1(z)r(z,a') - \theta w_2(z)c(z,a')\right)} = \mathbb{E}_{a\sim\widetilde{\pi}_{-\theta\beta}(\cdot|z)}w_2(z)c(z,a).$$

Table C.1: Dataset statistics and cost ratios.

Subset	Split	# Pairs	Cost Ratio
Magpie Ultra	Train 1	27,785	11.2
WildGuardMix	Train 2	6,709	3.4
OffsetBias	OOD Test	8,504	4.7

By (B.22), we further obtain,

$$\psi''(\theta) = (-\beta)\left(-\frac{w_2(z)^2}{\beta}\mathrm{Var}_{a\sim\widetilde{\pi}_\lambda(\cdot|z)}c(z,a)\right) = w_2(z)^2\,\mathrm{Var}_{a\sim\widetilde{\pi}_\lambda(\cdot|z)}c(z,a) \le M^2K^2.$$

By applying Taylor's expansion to ψ, there is

$$\psi(\theta_t) = \psi(\theta^*) + \psi'(\theta^*)(\theta_t-\theta^*) + \frac{1}{2}\psi''(\widetilde{\theta})(\theta_t-\theta^*)^2,$$

where $\widetilde{\theta}$ lies between θ_t and θ^*. Consequently, we get

$$D_\psi(\theta_t,\theta^*) = \frac{1}{2}\psi''(\widetilde{\theta})(\theta_t-\theta^*)^2 \le \frac{M^2K^2}{2}(\theta_t-\theta^*)^2 = \frac{M^2K^2}{2\beta^2}(\lambda_t-\lambda^*)^2,$$

which is equivalent to

$$\mathrm{KL}(\pi_t(\cdot|z)\|\pi^*(\cdot|z)) \le \frac{M^2K^2}{2\beta^2}(\lambda_t-\lambda^*)^2.$$

Averaging over $(z,l)\sim\rho$, we get

$$\mathrm{KL}(\pi_t\|\pi) = \mathbb{E}_{(z,l)\sim\rho}\mathrm{KL}(\pi_t(\cdot|z)\|\pi^*(\cdot|z)) \le \frac{M^2K^2}{2\beta^2}(\lambda_t-\lambda^*)^2.$$

Combining this with (B.23), we finally derive

$$\mathrm{KL}(\pi_t\|\pi) \le \frac{M^2K^2}{2\beta^2}\left(\frac{M^2K^2}{M^2K^2+2\beta^2}\right)^{2t}(\lambda_0-\lambda^*)^2.$$

C Experimental Details

C.1 Data Construction and Evaluation Protocol

We construct all judge data using a unified generation protocol, and explicitly separate datasets used for training and in-distribution analysis from those reserved for out-of-distribution (OOD) evaluation.

Judge Generation Protocol. Across all datasets, we generate judge responses using the same base model from the Qwen3 family (1.7B/4B/8B), under two inference modes: non-reasoning and reasoning. All generations use an identical prompting format (Table C.3) and a fixed sampling temperature of 0.6. This ensures that any observed performance or cost differences are attributable solely to the inference mode and data distribution.

Training and In-Distribution Data. We construct the training corpus from three sources: Skywork-Reward Liu et al. (2024a), Math-Step-DPO-10K Lai et al. (2024), and Code-Preference-Pairs Vezora (2024). Specifically, we sample 20,000 instances from Skywork-Reward, and an additional 20,000 instances from a mixture of Math-Step-DPO-10K and Code-Preference-Pairs. Together, these datasets cover general instruction-following, mathematical reasoning, and code-related preference judgments.

For each instance, we generate paired judge outputs under both inference modes. We record two signals for each mode: (i) a binary correctness indicator, obtained by comparing the judge's preference with the reference label provided by the dataset, and (ii) the number of tokens consumed during generation. These paired correctness and token-cost measurements define the empirical accuracy–cost trade-offs used both for the analysis in Section 2 and for training the routing policy. All training and in-distribution data are used exclusively for learning the routing policy.

Out-of-Distribution Evaluation Data. To evaluate generalization, we assess routing performance on multiple held-out benchmarks that are disjoint from the training data, including JUDGEBENCH Tan et al. (2024b), REWARDBENCH Lambert et al. (2025), and REWARDBENCH-2 Malik et al. (2025). These benchmarks span diverse evaluation domains and exhibit distributional shifts relative to the training corpus. We briefly summarize the domains they cover and the distribution in Table C.2.

For all OOD benchmarks, we apply the same judge generation protocol as used for training data, including the same base models, inference modes, prompts, and sampling temperature. Evaluation datasets are used solely for OOD testing and do not influence model selection or hyperparameter tuning.

C.2 TEXT REPRESENTATIONS

Each input instance is represented by a fixed-dimensional text embedding extracted via bge-3 (Chen et al., 2024a). Specifically, we encode the prompt and the two candidate responses separately, and concatenate their embeddings to form the policy input vector. The same representation procedure is applied consistently across all training and evaluation datasets.

C.3 ROUTING POLICY ARCHITECTURE AND OPTIMIZATION

Our routing policy is a 4-layer MLP with ReLU activations and hidden widths $\{256, 128, 64\}$, mapping the input embedding to a single scalar logit.

Models are trained for 60 epochs using the AdamW optimizer with learning rate 10^{-4} and batch size 64. The dual variable associated with the budget constraint is updated using a step size of 10^{-3} and projected onto the non-negative reals after each update. An entropy regularization term with coefficient 0.005 is applied to encourage exploration and stabilize training.

For model selection, we evaluate checkpoints on a held-out validation set. Among checkpoints that satisfy the target budget constraint, we select the one achieving the highest validation accuracy. If no checkpoint satisfies the constraint, we select the checkpoint whose validation cost is closest to the target budget.

C.4 BUDGET SETTINGS AND REPEATED TRIALS

We evaluate the routing policy under a range of target compute budgets:

$$B \in \{2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 6.0, 7.0, 10.0\}.$$

For each budget level, we repeat the entire training and evaluation procedure 10 times with independent random seeds. We report the mean of both accuracy and realized cost ratio across these runs.

C.5 HYPERPARAMETER TUNING

Optimization hyperparameters. We select standard optimization hyperparameters via a grid search on an in-distribution validation set, and then fix them for all experiments. This includes the regularization parameter β, the learning rate (10^{-4}), batch size (64), the dual update step size (10^{-3}), and the entropy regularization coefficient (0.005).

Temperature for robustness. Since we assume no access to supervision for validation for OOD evaluation, we adopt the following rule-of-thumb. Let τ_R and τ_C denote the robustness strengths for reward and cost, respectively, where smaller values impose more aggressive (more robust) reweighting, while larger values indicate a more non-robust behavior. If the anticipated an OOD dataset requires less computation than in-distribution, we recommend using a smaller τ_R (stronger reward robustness) and a larger τ_C (weaker cost robustness) to better utilize the budget. If the anticipated an OOD dataset require more computation, we recommend using a smaller τ_C (stronger cost robustness) to reduce budget-violation risk, while keeping τ_R moderate to avoid over-conservatism. When the shift direction is uncertain, a conservative default is to use moderate robustness on both terms.

In the OOD evaluation in Section 5.3, we fix $\tau_R = 1$ and disable robust reweighting on the cost term.

D SUPPLEMENTARY EXPERIMENTAL RESULTS

D.1 TRAINING CURVES

In Figure D.1, we report the training curves for reward, realized cost, and lambda under different budgets. These training curves indicate that the Lagrange multiplier dynamically adapts to the tightness of the compute budget: it grows under stringent budgets to actively enforce cost control, but collapses toward zero as the budget loosens, signaling that the constraint becomes non-binding.

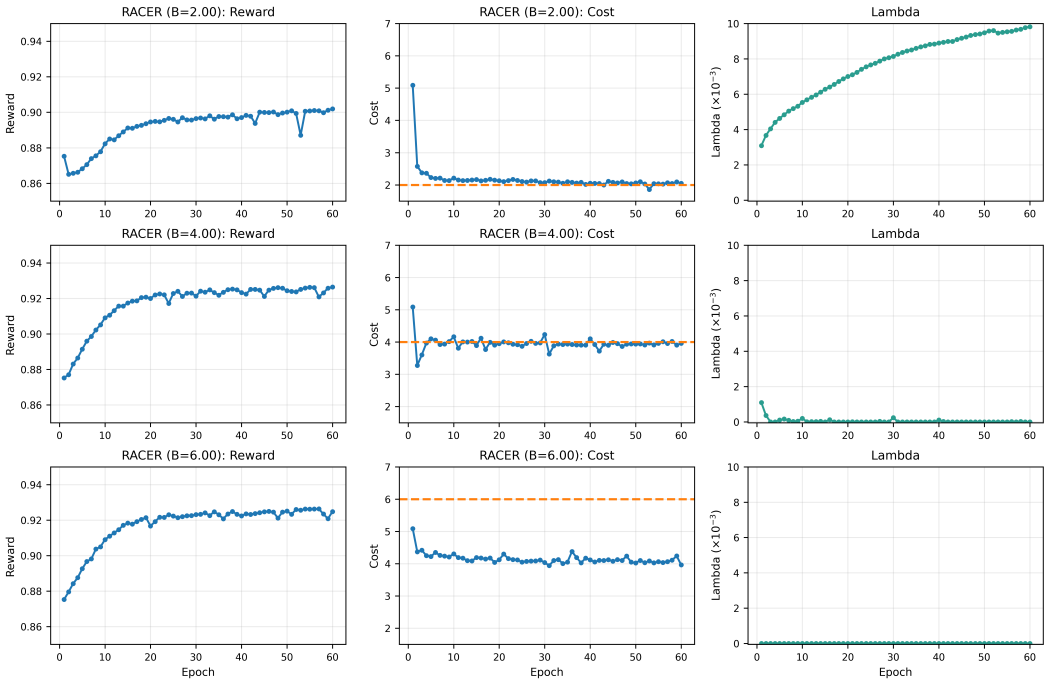


Figure D.1: Training dynamics of RACER under different cost budgets. Training reward (left), realized cost (middle), and the learned dual variable λ (right) across epochs for budgets $B \in \{2, 4, 6\}$. The dashed line denotes the target cost budget. Reward increases monotonically while λ adapts to the tightness of the constraint, driving the training cost toward the specified budget.

D.2 CASE STUDIES

In Table D.1, we report some representative cases to illustrate when and why reasoning helps.

Table C.2: Domain distribution of OOD evaluation datasets.

Benchmark	Domain	Count	Proportion	Description
JudgeBench Tan et al. (2024b)	Coding	73	11.8%	Challenging programming questions from contest-style platforms (e.g., LeetCode, AtCoder, Codeforces).
	Math	90	14.5%	Problems drawn from math competitions (e.g., AMC12, USAMO).
	Reasoning	149	24.0%	Reasoning-focused evaluation set.
	Knowledge	308	50.0%	College-level multiple-choice questions across 14 disciplines (e.g., Physics, Chemistry, Law), with up to 10 answer options.
RewardBench Lambert et al. (2025)	Math	447	15.0%	Aggregated from RewardBench original subsets. Subsets: *math-prm*.
	Coding	984	33.0%	Subsets: *hep-cpp*, *hep-go*, *hep-java*, *hep-js*, *hep-python*, *hep-rust*.
	Chat	814	27.3%	General assistant-style. Subsets: *alpacaeval-easy*, *alpacaeval-length*, *alpacaeval-hard*; *mt-bench-easy*, *mt-bench-medium*, *mt-bench-hard*; *llmbar-natural*, *llmbar-adver-neighbor*, *llmbar-adver-GPTInst*, *llmbar-adver-GPTOut*, *llmbar-adver-manual*.
	Safety	740	24.8%	Refusal/safety-oriented. Subsets: *refusals-dangerous*, *refusals-offensive*, *xstest-should-refuse*, *xstest-should-respond*, *do-not-answer*.
RewardBench 2 Malik et al. (2025)	Factuality	475	26.9%	Evaluates detection of hallucinations and other basic factual errors in completions.
	Precise IF	160	9.1%	Precise instruction following: judges whether outputs satisfy detailed, constraint-heavy instructions.
	Math	183	10.4%	Assesses math ability on open-ended prompts spanning middle-school topics through college-level chemistry, calculus, and combinatorics.
	Safety	450	25.5%	Measures appropriate compliance vs. refusal under harmful-use prompts and general safety-related behaviors.
	Focus	495	28.1%	Tests whether models prefer high-quality, on-topic answers for general user queries.

Table C.3: Judge prompting templates for `instruct` vs. `reasoning` modes. Differences are highlighted in red.

Field	Instruct mode	Reasoning mode
System	Please act as an impartial judge and evaluate the quality of the responses provided by two AI assistants to the user question displayed below. You should choose the assistant that follows the user's instructions and answers the user's question better. Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of their responses. Begin your evaluation by comparing the two responses and provide **a short explanation** explanation. Avoid any position biases and ensure that the order in which the responses were presented does not influence your decision. Do not allow the length of the responses to influence your evaluation. Do not favor certain names of the assistants. Be as objective as possible. After providing your explanation, output your final verdict by strictly following this format: ``[[A]]'' if assistant A is better, ``[[B]]'' if assistant B is better.	Please act as an impartial judge and evaluate the quality of the responses provided by two AI assistants to the user question displayed below. You should choose the assistant that follows the user's instructions and answers the user's question better. Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of their responses. Begin your evaluation by comparing the two responses and provide **an explanation**. Avoid any position biases and ensure that the order in which the responses were presented does not influence your decision. Do not allow the length of the responses to influence your evaluation. Do not favor certain names of the assistants. Be as objective as possible. After providing your explanation, output your final verdict by strictly following this format: ``[[A]]'' if assistant A is better, ``[[B]]'' if assistant B is better.
User template	[User Question]{question} [The Start of Assistant A's Answer]{answer_a}[The End of Assistant A's Answer] [The Start of Assistant B's Answer]{answer_b}[The End of Assistant B's Answer]	

Table D.1: Empirical case studies across benchmarks (excerpts only). Each row contrasts non-reasoning vs. reasoning judge outputs on the same instance; the label indicates whether the judge decision matches the ground-truth winner.

Dataset (ID)	Non-Reasoning Judge	Reasoning Judge
JUDGEBENCH-MATH (515)	**INCORRECT** (✗) *(excerpt)* Both Assistant A and Assistant B provide step-by-step solutions. The judge prefers Assistant B due to its more detailed explanations and justifications, which are considered more convincing and thorough, despite an error in counting the contribution of $49 = 7^2$. … Final verdict: `[[B]]`	**CORRECT** (✓) *(excerpt)* The judge explicitly verifies the computation using the Legendre formula: $\lfloor \frac{50}{7} \rfloor + \lfloor \frac{50}{49} \rfloor = 7 + 1 = 8$ It identifies that Assistant B undercounts the contribution of 49, and correctly determines that Assistant A provides the correct answer. … Final verdict: `[[A]]`
JUDGEBENCH-KNOWLEDGE (38)	**CORRECT** (✓) *(excerpt)* Both Assistant A and Assistant B provided thorough and accurate responses to the user's question. … Assistant B's response is more detailed and provides a step-by-step analysis of each option. They correctly identify the correct answers as (D), (G), and (I) but choose to focus on (D) as the main correct answer. … Additionally, Assistant B's response is more aligned with the user's request for a single correct answer. Therefore, based on the evaluation, I would say that Assistant B's response is slightly better than Assistant A's response. `[[B]]`	**INCORRECT** (✗) *(excerpt)* Okay, so I need to evaluate which AI assistant, A or B, provided a better answer … Assistant A's response … They provided explanations for each, pointing out which were correct and incorrect. For example, they correctly identified that option (I) is the right answer for vibration sense with a 128 Hz tuning fork. … Assistant B's response … However, their reasoning had some issues. For instance, they mentioned that the assessment should focus on the dominant arm … and they didn't fully explain why 128 Hz is better than 256 Hz for vibration testing. They also made a mistake by suggesting that the correct answer is (D) when the user's main question was about vibration sense, which should point to (I) instead. … Assistant B, while correct on some points, didn't address the primary question accurately and had some inaccuracies in their reasoning. Therefore, Assistant A's response is more comprehensive and accurate. `[[A]]`
RM-BENCH-CHAT (chat/404/0_0)	**CORRECT** (✓) *(excerpt)* Both Assistant A and Assistant B provide a concise and relevant answer. Assistant A correctly states that Mars appears reddish due to iron oxide … On the other hand, Assistant B incorrectly states that Mars appears reddish due to copper oxide … Considering the accuracy of the responses, Assistant A's answer is more reliable and trustworthy. … Therefore, based on the evaluation, the final verdict is: `[[A]]`	**CORRECT** (✓) *(excerpt)* Okay, let's see. I need to evaluate the responses … Assistant A's response … It's concise and accurate. Assistant B's response … Both responses mention reddish and rust, but Assistant B says copper oxide instead of iron oxide. That's a mistake. … So, in terms of accuracy, Assistant A is correct, whereas Assistant B is wrong. … However, the accuracy is key here. Since Assistant A provided the right information, it's better. … `[[A]]`