Aligning Language Models with Selective Prediction

Gaoxiang Luo[1], **Yifan Wu**[2], **Sinian Zhang**[3], **Aryan Deshwal**[1], **Ju Sun**[1]
[1]Department of Computer Science and Engineering
[2]Bioinformatics and Computational Biology Program
[3]Division of Biostatistics and Health Data Science
University of Minnesota Twin Cities
{luo00042,wu001810,zhan9381,adeshwal,jusun}@umn.edu

Project website: https://sun-umn.github.io/RLSR

Abstract

Large language models (LLMs) are increasingly deployed as critical decision-making components in high-stakes real-world AI systems, rendering LLM reliability a foremost practical concern. In this paper, we focus on enhancing LLM reliability through selective prediction (SP), a strategy that allows an LLM to only predict for inputs where it is likely to be correct (i.e., *coverage*) and hence reduce the error rate (i.e., *risk*) on that portion of inputs—flagging the remaining inputs for future human discretion. In other words, SP improves LLM reliability by balancing the risk-coverage trade-off and enabling seamless human-AI collaboration. To integrate SP into LLMs, we focus on the LLM post-training alignment stage and propose to align LLMs with SP performance metrics, in contrast with existing LLM alignment methods that focus primarily on correctness or calibration metrics. Specifically, we propose a novel alignment framework, Reinforcement Learning for Selection Reward (RLSR), which targets the area under the risk-coverage curve (AURC)—a popular SP performance metric—as its alignment objective. RLSR achieves substantially better risk-coverage trade-off compared to multiple alignment baselines on both in-domain and out-of-domain tasks.

1 Introduction

In high-stakes applications such as finance, healthcare and law [1–3], LLMs are increasingly used to make decisions. In these settings, the cost of making a mistake significantly outweighs that of deferring the case to a human. Thus, it is reasonable to allow and encourage the LLM to abstain when its answer is likely to be wrong. This is the goal of **selective prediction (SP)**, where the LLM answers only a selected subset of queries that it is likely to get right, to keep its error rate acceptably low on that subset. This paradigm stands in stark contrast to standard LLM training and evaluation, which requires LLMs to answer regardless of their capabilities, causing confident mistakes and hallucinations [4]. In this work, we propose aligning LLMs directly with SP in a principled manner.

Confidence calibration (CC) is not sufficient or necessary for selective prediction (SP). SP is often assumed, *explicitly or implicitly*, to be the logical next step after confidence calibration (CC) [5–9]. Indeed, assume that the prediction confidence is perfectly calibrated (i.e., aligned with the true posterior probability $p(y|x)$); then a selection threshold at 0.8 ensures that the error rate on the selected subset is less than $1 - 0.8 = 0.2$. While this is technically elegant and convenient, CC and SP are still disparate: CC stresses that the assigned confidence be consistent with the likelihood of correctness, whereas SP favors that correct predictions are ranked ahead of wrong ones—in terms of assigned confidence. As shown in Fig. 1 (Left), *perfect CC might not lead to perfect SP*, as any threshold we draw can still lead to accepted wrong answers and rejected right answers; conversely, as

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

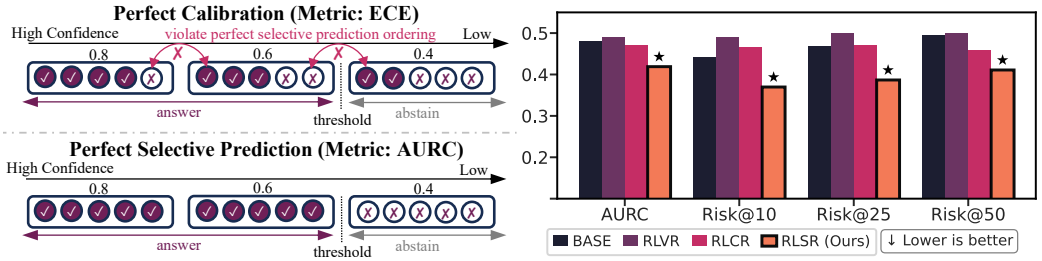


Figure 1: **(Left)** Perfect confidence calibration does not translate to perfect selective prediction, and vice versa; **(Right)** Our RLSR aligns LLMs with selective prediction (SP) and substantially improves upon competing methods in terms of SP metrics: AURC and $\mathrm{Risk@}\,k$.

shown in Fig. 1 (Right), *perfect SP can be based on ill-calibrated confidence*, so long as all right answers are ranked ahead of wrong ones. Their difference is also reflected by the different evaluation metrics used in practice: CC is often measured by the *expected calibration error* (ECE) or variants, whereas SP by the *area under the risk-coverage curve* (AURC) or variants. Thus, *if the goal is SP, it is technically sound to pursue it directly instead of through CC*.

Aligning LLMs with selective prediction. In this paper, we propose aligning pretrained LLMs directly with SP. To this end, we adopt the popular Reinforcement Learning with Verifiable Rewards (RLVR) [10] alignment framework, which uses verifiable rewards to improve LLM accuracy and reasoning. The recent work, Reinforcement Learning with Calibration Rewards (RLCR) [11], proposes augmenting the correctness reward in RLVR with CC-aware rewards (e.g., the Brier score) to promote LLM accuracy and CC. However, it does not necessarily optimize SP for LLMs, due to the misalignment of CC and SP discussed above. Thus, we propose replacing the correctness reward with SP-aware rewards. Toward this, the AURC, a standard SP metric that summarizes the whole spectrum of risk-coverage tradeoffs, seems a natural choice. However, we face two implementation and performance challenges: (1) the canonical AURC only penalizes errors but does not reward correctness, impairing learning; (2) AURC is a population-level metric that depends on the global ranking of sample confidence, not friendly for mini-batch stochastic gradient methods.

Our contributions. In this paper, we propose **Reinforcement Learning for Selection Reward (RLSR)** to directly align pretrained LLMs with SP. To tackle the challenges identified above, we develop a lifted AURC objective, and a batch training strategy based on a judicious choice of AURC reformulation. Our key contributions are:

1. We propose aligning pretrained LLMs with selective prediction (SP) to enhance their reliability for deployment. To the best of our knowledge, this is *the first work* proposing this;
2. We introduce RLSR, a new LLM alignment framework with an SP-aware reward. We base our learning algorithm on the Group Relative Policy Optimization (GRPO) framework [10] and develop a novel lifted AURC objective and a batch training strategy to tackle the key technical challenges in optimizing the AURC metric (Sec. 2.2);
3. We empirically demonstrate that RLSR outperforms strong baselines, including accuracy-oriented RLVR and calibration-oriented RLCR, achieving strong SP performance on multiple in-domain and out-of-domain tasks (Fig. 1 (Right), Secs. 4.1 and 4.2).

2 Method

2.1 Technical Background

LLM alignment via RL. Let $\mathcal{X}$ and $\mathcal{Y}$ be the prompt and response spaces, respectively, and $\mathcal{D}$ be the data distribution on $\mathcal{X} \times \mathcal{Y}$. Given a stochastic LLM π_θ that maps prompts to responses, a typical LLM alignment step is to improve LLM responses by optimizing

$$\max_\theta J(\theta) \doteq \mathbb{E}_{(x,y)\sim\mathcal{D}} \mathbb{E}_{\widehat{y}\sim\pi_\theta(\cdot|x)} R(y, \widehat{y}), \tag{2.1}$$

where $\pi_\theta(\cdot|x)$ denotes the distribution on the response space $\mathcal{Y}$ conditioned on a prompt x—not a single response because π_θ is stochastic, and $R : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is a chosen **reward function**. In other words, LLM alignment is to finetune the LLM parameters θ by maximizing the expected reward. To

maximize using gradient ascent, one can apply the celebrated log-derivative trick, which is popularly used in policy gradient methods for reinforcement learning (RL), to derive the expected gradient as (see Sec. A.1 for details)

$$\nabla_\theta J(\theta) = \mathbb{E}_{(x,y)}\mathbb{E}_{\widehat{y}}\left[\nabla_\theta \log \pi_\theta(\widehat{y}|x) R(y,\widehat{y})\right], \tag{2.2}$$

where we omit the $\mathcal{D}$ and $\pi_\theta(\cdot|x)$ under expectation signs to simplify the notation. In practice, the expected gradient is approximated by mini-batch gradient: one estimates the gradient by sampling a batch of (x, y) pairs from $\mathcal{D}$, and for each x sampling a batch of $\widehat{y}$, i.e., rollouts, from $\pi_\theta(\cdot|x)$.

In practice, various techniques have been proposed to improve the performance of the above framework. We follow the popular **GRPO** algorithm [10], which is essentially the classic **proximal policy optimization (PPO)** in RL with a twist on the advantage function. PPO changes the optimization objective in Eq. (2.1) in three aspects: **(1) rewards into group-relative advantages** by replacing $R(y,\widehat{y})$ with prompt-level advantage $A(y,\widehat{y})$ that gives more advantages to "breakthrough" responses that stand out from the rest; **(2) per-iteration surrogation** by approximating the objective by

$$J(\theta;\theta_{\text{old}}) \doteq \mathbb{E}_{(x,y)}\mathbb{E}_{\widehat{y}\sim\pi_{\theta_{\text{old}}}(\cdot|x)} \frac{\pi_\theta(\widehat{y}|x)}{\pi_{\theta_{\text{old}}}(\widehat{y}|x)} A(y,\widehat{y}), \tag{2.3}$$

where θ_{old} denotes the model parameters from the previous iteration. This helps decouple response sampling from an old model $\pi_{\theta_{\text{old}}}$ and gradient estimation for the current model π_θ; **(3) range clipping on density ratios** to stabilize training by modifying Eq. (2.3) as

$$r_{x,\widehat{y}} \to \min(1+\varepsilon, r_{x,\widehat{y}})\mathbb{I}\{A(y,\widehat{y})>0\} + \max(1-\varepsilon, r_{x,\widehat{y}})(1-\mathbb{I}\{A(y,\widehat{y})>0\}) \tag{2.4}$$

where $r_{x,\widehat{y}} \doteq \pi_\theta(\widehat{y}|x)/\pi_{\theta_{\text{old}}}(\widehat{y}|x)$, i.e., avoid excessively positive and negative advantages so that the update to θ becomes less biased toward any particular $(x,\widehat{y})$ pair. GRPO, as a variant of PPO, works with the special advantage function

$$A_{\text{GRPO}}(y,\widehat{y}_i) = \left[R(y,\widehat{y}_i) - \text{mean}\left\{R(y,\widehat{y}_j)\right\}_{j=1}^G\right] / \text{std}\left\{R(y,\widehat{y}_j)\right\}_{j=1}^G, \tag{2.5}$$

where j indexes the sampled responses (i.e., rollouts) for the same prompt. This advantage function effectively performs prompt-level zero-mean-unit-variance normalization to the rewards. The gradient of the modified objective in GRPO can be derived using the log-derivative trick, analogous to the process for Eq. (2.1).

RLVR & RLCR. RLVR has recently emerged as the gold-standard LLM alignment framework for mathematical and coding tasks [12, 13]. It typically assumes an external programmatic verifier to judge the correctness of the response and gives a binary reward

$$R_{\text{RLVR}}(y,\widehat{y}) \doteq \mathbb{I}\{y=\widehat{y}\}. \tag{2.6}$$

After RLVR training, recent reasoning models often return verbalized confidence [14, 15] alongside their responses, explicitly communicating the model's internal uncertainty to the user (see Sec. C.1). Although R_{RLVR} explicitly incentivizes accurate prediction, RLVR-finetuned LLMs are often found to be overconfident, i.e., not well calibrated [16]. To promote both correct prediction and calibrated confidence, RLCR [11] proposes adding a Brier score penalty to the RLVR reward, leading to

$$R_{\text{RLCR}}(y,\widehat{y},s) \doteq \mathbb{I}\{y=\widehat{y}\} - \left(s - \mathbb{I}\{y=\widehat{y}\}\right)^2, \tag{2.7}$$

where s denotes the verbalized confidence returned by the LLM alongside $\widehat{y}$.

Selective prediction (SP). For a general input-output space $\mathcal{X}' \times \mathcal{Y}'$ and an associated data distribution $\mathcal{D}'$ on $\mathcal{X}' \times \mathcal{Y}'$, SP augments a predictive model $f : \mathcal{X}' \to \mathcal{Y}'$ with a binary selector $g : \mathcal{X}' \to \{0,1\}$, so that the final output by the pair (f,g) is:

$$(f,g)(x') = \begin{cases} f(x') & \text{if } g(x')=1, \\ \text{abstain} & \text{if } g(x')=0 \end{cases} \quad \forall x' \in \mathcal{X}'. \tag{2.8}$$

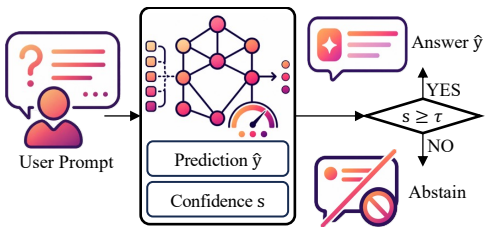


Figure 2: SP for LLMs in deployment.

The selector g typically takes the form

$$g_{s,\tau}(x') \doteq \mathbb{I}\{s(x') \geq \tau\}, \tag{2.9}$$

where $s : \mathcal{X}' \to [0,1]$ is a confidence scorer and $\tau \in [0,1]$ is a selection threshold. The selector chooses to keep only the most confident predictions based on the confidence scorer s with a confidence cutoff τ. To evaluate the performance of SP, two metrics are of particular interest: the **coverage** is the portion of predictions *effectively* covered by the selector, i.e., not rejected by the selector

$$\phi_{s,\tau}(f) \doteq \mathbb{E}_{(x',y')\sim\mathcal{D}'}[g_{s,\tau}(x')]\,; \tag{2.10}$$

the **selective risk** is the risk over the covered subpopulation

$$R_{s,\tau}(f) \doteq \mathbb{E}_{(x',y')\sim\mathcal{D}'}[\mathbb{I}\{f(x') \neq y\}\, g_{s,\tau}(x')]/\phi_{s,\tau}(f), \tag{2.11}$$

where the binary loss $\mathbb{I}\{f(x') \neq y\}$ could be replaced by other appropriate losses. For any fixed (f, s), there is typically a tradeoff between coverage and selective risk determined by τ. So, one can vary τ over $[0,1]$ to plot the entire tradeoff curve, called the **risk-coverage curve**. A popular summary metric for quantifying SP performance is the **area under the risk-coverage curve** (AURC). For a finite dataset of size N, the coverage varies by $1/N$ (assuming that there is no tie in the confidence). So, the empirical AURC is computed as

$$\overline{\mathrm{AURC}} \doteq \frac{1}{N}\sum\nolimits_{i=1}^{N} R_{(i)}, \quad \text{where } R_{(i)} \;=\; \frac{1}{i}\sum\nolimits_{j=1}^{i} \mathbb{I}\{f(x_{(j)}) \neq y_{(j)}\} \tag{2.12}$$

i.e., $R_{(i)}$ denotes the selective risk computed on the subset of the most confident samples i (the subscript $\cdot_{(j)}$ indicates the sample with the j-th highest confidence score). Clearly, many of the indicator terms are repeated across different $R_{(i)}$'s. The equivalent form of [17] collects the repeated indicator terms together so that each sample appears inside the summation exactly once, i.e., as a weighted empirical loss

$$\overline{\mathrm{AURC}}^{\mathrm{w}} = \frac{1}{N}\sum\nolimits_{i=1}^{N} \widehat{\alpha}_i \cdot \mathbb{I}\{f(x_i) \neq y_i\}, \quad \text{with } \widehat{\alpha}_i = H_N - H_{N-r_i} = \sum\nolimits_{j=1}^{r_i} \tfrac{1}{N-j+1}. \tag{2.13}$$

Here, r_i denotes the ranking of the i-th sample in ascending order of confidence, and H_N represents the N-th harmonic number. This weighted form is our primary focus below thanks to its conciseness.

2.2 Our Method: Reinforcement Learning for Selection Reward (RLSR)

As discussed in Sec. 1, although RLVR and RLCR could improve LLM accuracy and perhaps also confidence calibration, they are not sufficient to boost LLM SP performance. Our goal here is to directly align pretrained LLMs with SP based on the GRPO framework described in Sec. 2.1. To implement GRPO, it is natural to design the reward to be consistent with $\overline{\mathrm{AURC}}^{\mathrm{w}}$ defined in Eq. (2.13). For SP, we choose to work with **verbalized confidence** as an exemplar confidence, following RLCR [11] and due to its simplicity and recent popularity [14, 15, 18, 19].

Two challenges with AURC. The negative version of $\overline{\mathrm{AURC}}^{\mathrm{w}}$ in Eq. (2.13) can serve as a natural reward at the population-level. However, two fundamental challenges arise when we try to integrate it into GRPO:

(1) **Sparse rewards.** $\overline{\mathrm{AURC}}^{\mathrm{w}}$ is a weighted sum of binary indicators of the form $\mathbb{I}\{f(x_i) \neq y_i\}$, which produces sparse one-sided learning signals with two deficiencies: **(i) No signals for correct predictions.** Since $\overline{\mathrm{AURC}}^{\mathrm{w}}$ only penalizes errors and does not assign learning signals to correct predictions, this could slow down learning or even impair it; **(ii) No margin enforcement.** Since correct predictions are not rewarded regardless of their confidence levels, a confident correct prediction and a lucky guess are treated equally. This would allow the confidence margin between correct and wrong predictions to be arbitrarily close—the alignment process has no incentives to promote it, which can in turn hurt generalization performance;

(2) **Rewards based on ranking.** The definition of $\overline{\mathrm{AURC}}^{\mathrm{w}}$ in Eq. (2.13) depends on $\widehat{\alpha}_i$, which depends on the ranking of the i-th sample over the entire dataset. This means that $\overline{\mathrm{AURC}}^{\mathrm{w}}$ is a population-level metric that depends on the global ranking of all samples. However, mini-batch stochastic gradient descent (SGD) applied to the GRPO framework clearly relies on mini-batch rewards in each step, which cannot be decoupled from the global ranking. But global reranking per iteration step would defeat the purpose of mini-batch SGD in avoiding computation over the whole dataset.

Lifted AURC to address the first challenge. We introduce a reformulation called *Lifted AURC*. The idea is to shift the binary indicator $\mathbb{I}\{\widehat{y}_i \neq y_i\} \in \{0,1\}$ to a signed variable $2\mathbb{I}\{\widehat{y}_i \neq y_i\} \in \{0,1\} - 1 \in \{-1,+1\}$ directly on Eq. (2.13), resulting in

$$\overline{\mathrm{AURC}}^{\mathrm{w}}_{\mathrm{lift}} = \frac{1}{N}\sum\nolimits_{i=1}^{N} \widehat{\alpha}_i \cdot \big(2\mathbb{I}\{\widehat{y}_i \neq y_i\} - 1\big). \tag{2.14}$$

Since $\frac{1}{N}\sum_{i=1}^{N} \widehat{\alpha}_i \cdot \big(2\mathbb{I}\{\widehat{y}_i \neq y_i\} - 1\big) = \frac{2}{N}\sum_{i=1}^{N} \widehat{\alpha}_i \cdot \mathbb{I}\{\widehat{y}_i \neq y_i\} - \frac{1}{N}\sum_{i=1}^{N} \widehat{\alpha}_i = \frac{2}{N}\sum_{i=1}^{N} \widehat{\alpha}_i \cdot \mathbb{I}\{\widehat{y}_i \neq y_i\} - 1$, we have $\overline{\mathrm{AURC}}^{\mathrm{w}}_{\mathrm{lift}} = 2\overline{\mathrm{AURC}}^{\mathrm{w}} - 1$. So, minimizing $\overline{\mathrm{AURC}}^{\mathrm{w}}_{\mathrm{lift}}$ is equivalent to minimizing $\overline{\mathrm{AURC}}^{\mathrm{w}}$.[1] This lifted version $\overline{\mathrm{AURC}}^{\mathrm{w}}_{\mathrm{lift}}$ provides two-sided rewards that actively reward correct predictions and penalize incorrect ones. The opposite signs of the nonzero rewards for correct and wrong predictions, respectively, create a "push-pull" dynamic that drives correct and wrong predictions apart in their confidence rankings, thereby improving the confidence margin. This margin enforcement is critical for SP, which relies on a sufficient separation of confidence between correct and incorrect predictions. In Fig. 3 (Left), we empirically confirm the margin benefit of using the

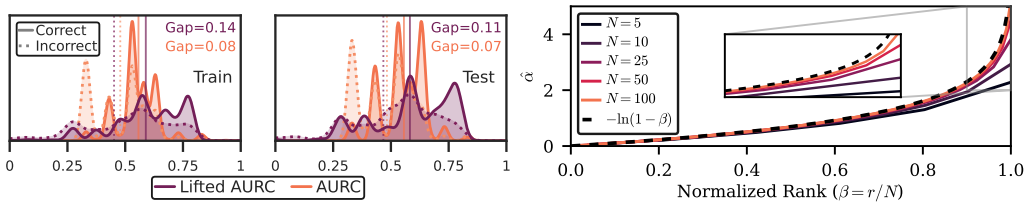


Figure 3: **(Left)** Normalized confidence distributions of correct and incorrect predictions. Empirically, lifted AURC as a reward achieves larger separation (margin) between correct and incorrect predictions on both train and test sets (see Sec. C.7); **(Right)** Behavior of the rank-based weights $\widehat{\alpha}$.

lifted AURC as a reward, compared to the plain AURC. Consistent with $\overline{\mathrm{AURC}}^{\mathrm{w}}_{\mathrm{lift}}$, we define the per-sample RLSR reward as

$$R_{\mathrm{RLSR}}(y_i, \widehat{y}_i, s_i) = \begin{cases} +\widehat{\alpha}_i & \text{if } \widehat{y}_i = y_i \\ -\widehat{\alpha}_i & \text{if } \widehat{y}_i \neq y_i \end{cases}. \tag{2.15}$$

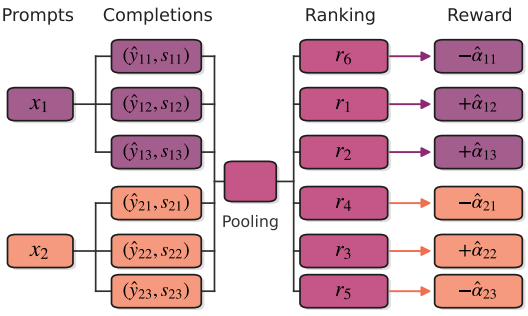


Figure 4: **Illustration of implementation.** We pool rollouts from B inputs, each generating G samples, into an effective batch of size $B \times G$ (e.g., $B = 2$, $G = 3$). Rewards are computed based on ranking over the effective batch. The subsequent relative advantage computation is identical to standard GRPO within each input group.

To confirm that this definition of the per-sample reward makes intuitive sense, in Fig. 3 (Right), we plot the weights $\widehat{\alpha}$ against the normalized rank $\beta = r/N$ (in ascending order of confidence) for varying batch sizes $N \in \{5, \ldots, 100\}$. The empirical weights converge to the theoretical curve $-\log(1-\beta)$ [17]. Moreover, this weighting scheme places much more emphasis on high-confidence predictions than on low-confidence ones (the asymptotic slope of the weight curve moving from low-confidence to high-confidence is $1/(1-\beta)$). This is sensible, as high-confidence ones are likely to persist in the selected subset when one gradually increases the selection threshold τ, than low-confidence ones that tend to be rejected early. Thus, attending more to the high-confidence ones helps improve the overall SP performance, aligned with our goal.

Batch training to address the second challenge. Although R_{RLSR} provides a reasonable reward per-sample, computing the weight $\widehat{\alpha}_i$ still requires ranking across all samples. We address this by the following batch training strategy (see Fig. 4). For each mini-batch of B samples, we independently generate G rollouts for each sample and then pool all these $B \times G$ rollouts into an effective batch.

[1]The equivalence is in the sense that they have the same set of global solutions. But due to the significant overparameterization of LLMs, the global solution is unlikely to be unique and they may have different levels of generalization.

We then compute ranks r_i and weights $\widehat{\alpha}_i$ across this pooled set, approximating the population-level AURC. The batch-level ranking step introduces negligible overhead and remains compatible with standard GRPO throughout. The pseudocode of the entire algorithm can be found in Alg. 1.

Comparison with RLVR. When rollouts for an input are all correct or wrong, RLVR assigns the same reward to all, yielding zero advantage and no learning signal. However, this *sparse reward* issue is less likely for RLSR because it differentiates rollouts by confidence ranks, providing a unique reward to each rollout (if no ties or ties are broken by averaging $\widehat{\alpha}_i$). This allows the model to separate low-confidence lucky guesses from high-confidence predictions and penalize confident errors more heavily than uncertain mistakes.

Algorithm 1 Our algorithm: GRPO with R_{RLSR} in Eq. (2.15)

Require: LLM π_θ, data $\mathcal{D}$, generations G, clipping ε, regex parser $\mathcal{P}$ for answer and verbalized confidence

for step $= 1, \ldots, T$ **do**
  Sample batch $\{(x_b, y_b)\}_{b=1}^{B}$ from $\mathcal{D}$
  $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$
  for $b = 1, \ldots, B$ **do**
    Sample G outputs from $\pi_{\theta_{\text{old}}}(\cdot \mid x_b)$
    $(\widehat{y}_{b,g},\ s_{b,g}) \leftarrow \mathcal{P}(\pi_{\theta_{\text{old}}}(\cdot \mid x_b))$ for $g = 1, \ldots, G$
  end for
  // Pool $n{=}B{\times}G$ rollouts for batch-level ranking
  Sort all n samples ascendingly; assign $r_i \in \{1, \ldots, n\}$
  $\widehat{\alpha}_i \leftarrow H_n - H_{n-r_i}$ {where $H_k \doteq \sum_{j=1}^{k} 1/j$}
  $R_i \leftarrow \begin{cases} +\widehat{\alpha}_i & \widehat{y}_i = y_i \\ -\widehat{\alpha}_i & \widehat{y}_i \neq y_i \end{cases}$
  // Group-relative advantages (GRPO)
  for $b = 1, \ldots, B$ **do**
    $A_{b,g} \leftarrow (R_{b,g} - \text{mean}_g(R_{b,\cdot}))\,/\text{std}_g(R_{b,\cdot})$
  end for
  // Clipped policy update (per-token)
  $\rho_{i,t} \leftarrow \pi_\theta(o_{i,t} \mid x, o_{i,<t})\,/\,\pi_{\theta_{\text{old}}}(o_{i,t} \mid x, o_{i,<t})$
  $\mathcal{L} \leftarrow -\mathbb{E}_{i,t}\big[\min\big(\rho_{i,t}A_i,\ \text{clip}(\rho_{i,t}, 1{\pm}\varepsilon)A_i\big)\big]$
  Update π_θ by $\nabla_\theta \mathcal{L}$
end for
return π_θ

3 Related Work

Confidence scorers for LLMs. In the SP literature, confidence scorers are often derived from prescribed rules (e.g., these based on output logits) or learned confidence predictors [8]. For LLMs, rule-based confidence scorers proposed in the literature are numerous, but the community has not converged to a standard [20]. Recent work [21, 22] has also developed learned confidence scorers for LLMs. Our RLSR framework is compatible with both types of confidence scorers, as our reward only requires a confidence scorer that ranks the samples alongside a correctness verifier.

Direct learning with AURC. In machine learning, learning with the AURC objective is still rare. Recently, [17, 23] has explored this for classification problems, but their focuses are mostly on the effective computation and approximation of AURC for learning and optimization purposes; [9] learns the ensemble of post-hoc scorer given a set of base scorers, toward optimizing the AURC loss. Our method in Sec. 2.2 hinges on the key equivalent definition of AURC as weighted empirical risk (i.e., Eq. (2.13)) developed in [17]. However, the two technical challenges we address in Sec. 2.2 are unique to the GRPO framework (and RL problems in general) and our RLSR algorithm, and therefore not considered in [17, 23].

We include more discussion of remoter related work in Sec. B.

4 Experiments

To verify the efficacy of our RLSR alignment framework, we benchmark RLSR against RLVR and RLCR in terms of risk-coverage tradeoff on both in-domain and out-of-domain tasks in Sec. 4.1. We further evaluate these alignment frameworks under a risk-controlled deployment setting in high-stakes domains in Sec. 4.2.

4.1 Benchmarking in risk-coverage tradeoff

Models and datasets. We consider both Qwen2.5-7B [24] and Llama-3.1-8B [25] as our base LLM models. We perform alignment training on two datasets: HotPotQA-Modified [26] and BigMath [27]. HotPotQA-Modified is a multi-hop question answering dataset that tests reasoning under incomplete or distracting evidence, while BigMath is a multi-step mathematical reasoning dataset where uncertainty accumulates across steps. *To diversify the evaluation sets and cover*

different uncertainty types other than those in the training datasets, we include 6 *additional datasets:* SimpleQA and TriviaQA [28, 29] probe overconfidence on rare factual knowledge; GPQA, Math500, and GSM8K [30–32] assess multi-step reasoning; and CommonsenseQA [33] examines ambiguous, implicit reasoning scenarios. To construct the in-domain (ID) and out-of-domain (OOD) evaluation sets, we organize the datasets as follows: for HotPotQA-aligned models, ID evaluation only uses HotPotQAs, and OOD evaluation includes the remaining 6 additional datasets; for BigMath-aligned models, Math500, GSM8K, and BigMath are for ID evaluation, while the rest are for OOD evaluation.

Competing methods, confidence scorers, and evaluation metrics. **Competing methods**. The key baseline methods we include are (1) BASE, no alignment, (2) RLVR, the standard correctness-only alignment, and (3) RLCR, the recent correctness+calibration alignment. We also compare against non-alignment alternatives such as supervised finetuning in Sec. D.6; **Confidence scorers**. To elicit verbalized confidence and reasoning about uncertainty, we include a format reward that encourages the model to output an analysis and a confidence score after the chain-of-thought answer (see Sec. C.4.2). Since our method is agnostic to the choice of the confidence scorer, we also experiment with other post-hoc confidence scorers for the selector; RLSR attains the best or tied-best AURC under every scorer (see Sec. D.7); **Evaluation metrics**. Our primary metric is the standard SP metric AURC, which directly measures the holistic risk-coverage tradeoff. Alongside AURC, we also report the accuracy (i.e., $1-$risk) at the 10%, 25%, and 50% coverage levels. High-stakes applications typically require operating in the low-risk (i.e., high-accuracy) regime, which corresponds to low coverage. For completeness, we also report ECE, the standard calibration metric. However, note that ECE is not a direct measure of SP performance and is not optimized by RLSR.

Training and inference details. For each prompt, we generate responses with temperature $T = 0.7$ and use an effective batch size of $B \times G = 1536$ samples. We then rank the samples inside the effective batch by their confidence scores to compute the RLSR reward (Alg. 1). To elicit structured outputs, we adapt the RLCR system prompts for HotPotQA and BigMath, which induce `<think>`, `<answer>`, `<analysis>`, and `<confidence>` tags. The maximum response length is 1536 tokens for HotPotQA and 4096 tokens for BigMath. All competing methods share identical training configurations and operate under equal inference budgets for fair comparison; full hyperparameter details are in Sec. C.4. Following [34], we omit the std normalization when calculating the GRPO advantage; an ablation study confirms the difference is minor (HotPotQA ID AURC: 0.46 with std normalization vs. 0.44 without)). In evaluation, all results use greedy decoding ($T = 0$) for reproducibility; statistical significance over 5 seeds at $T = 0.3$ is reported in Sec. D.2, and robustness to the choice of LLM judge in Sec. D.3.

Table 1: Comparison of SP performance on in-domain (ID) (2 for HotPotQA and 3 for BigMath) and out-of-domain (OOD) evaluation datasets (6 for HotPotQA and 5 for BigMath) with the Qwen2.5-7B model. Orange highlights the best result per metric within each training setting. RLSR also outperforms all baselines on the Llama-3.1-8B model; see Tab. 4.

	In-Domain					Out-of-Domain				
Method	AURC↓	ECE↓	Acc@10↑	Acc@25↑	Acc@50↑	AURC↓	ECE↓	Acc@10↑	Acc@25↑	Acc@50↑
HotPotQA Training										
BASE	0.60	0.57	46.0%	40.6%	37.8%	0.44	0.40	59.1%	57.4%	54.9%
RLVR	0.56	0.56	45.5%	42.0%	41.8%	0.47	0.46	52.6%	52.9%	53.1%
RLCR	0.51	0.05	53.5%	49.0%	50.4%	0.46	0.23	53.5%	54.3%	55.3%
RLSR	0.44	0.24	66.0%	63.2%	57.2%	0.41	0.28	62.0%	60.7%	59.5%
BigMath Training										
BASE	0.41	0.40	58.9%	61.5%	57.9%	0.49	0.44	54.7%	51.1%	49.9%
RLVR	0.28	0.25	72.2%	70.7%	71.5%	0.53	0.49	46.4%	43.9%	46.7%
RLCR	0.26	0.12	76.7%	75.5%	73.6%	0.47	0.21	57.3%	53.8%	52.1%
RLSR	0.24	0.18	78.8%	78.4%	76.4%	0.45	0.34	58.5%	56.3%	54.2%

RLSR's superior in-domain performance. From Tab. 1, it is evident that RLSR consistently outperforms RLVR and RLCR in terms of both AURC and accuracy-at-coverage across all three coverage levels for ID evaluation. For example, for models trained on HotPotQA, RLSR achieves an AURC of 0.44 compared to 0.56 for RLVR and 0.51 for RLCR; and at 10% coverage, RLSR reaches 66.0% accuracy versus 45.5% for RLVR and 53.5% for RLCR. Models trained on BigMath exhibit

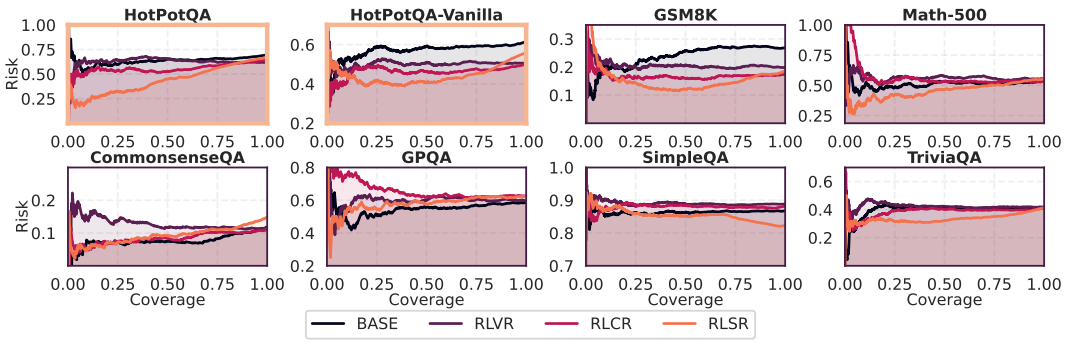


Figure 5: The risk-coverage curves of the Qwen2.5-7B model aligned on HotPotQA. The in-domain datasets are framed in orange while the remaining are out-of-domain datasets.

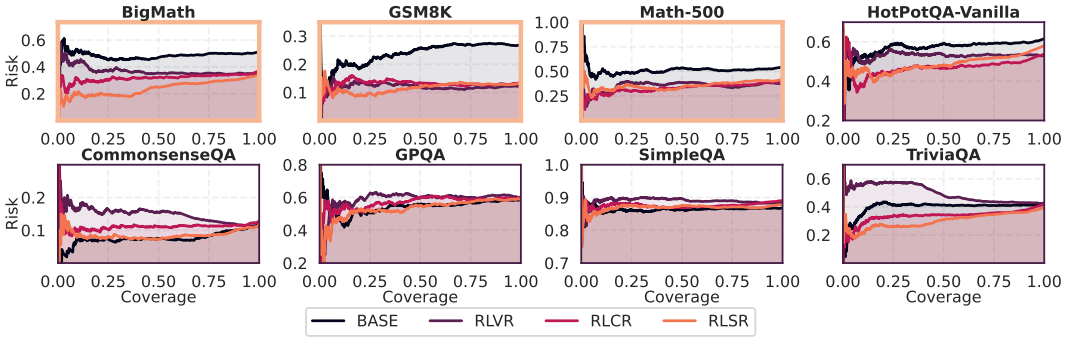


Figure 6: The risk-coverage curves of the Qwen2.5-7B model aligned on BigMath. The in-domain datasets are framed in orange while the remaining are out-of-domain datasets.

similar gains across all metrics. Notably, RLCR achieves the best ECE but lags behind RLSR in AURC, whereas RLSR has higher ECE than RLCR but the best AURC. This provides direct empirical evidence for the misalignment between calibration and SP discussed in Sec. 1 and Fig. 1 (Left): better calibration does not imply better SP, and vice versa. So, RLSR is the better choice when selection is the goal.

RLSR's SP capability transferable to novel domains. RLSR's SP performance remains strong on OOD evaluation, as shown in Tab. 1. It uniformly improves over all baseline methods, and particularly by large margins on the HotPotQA group. Surprisingly, RLVR always degrades the OOD SP performance relative to the base method without any alignment, and RLCR degrades the OOD SP performance in HotPotQA, but improves the performance in BigMath. We suspect that such performance degradation is partially due to overfitting the ID distribution during the alignment process, which makes the model less robust to distribution shifts. Notably, RLSR's SP performance advantage remains consistent despite the distribution shifts. The advantage also extends to code generation, a task type absent from both training sets: on MBPP [35], RLSR attains the best AURC, Acc@10, and Acc@25 for both training domains (Sec. D.4).

Table 2: Controlled confidence gap analysis on HotPotQA ID (Qwen2.5-7B). We restrict to cases where all 3 methods predict correctly or incorrectly, controlling accuracy as a confounder. $\Delta c = \mathbb{E}[c|\text{correct}] - \mathbb{E}[c|\text{incorrect}]$ measures how well each method separates correct from incorrect predictions by confidence.

Method	$\mathbb{E}[c\|\text{correct}]$	$\mathbb{E}[c\|\text{incorrect}]$	$\Delta c \uparrow$
RLVR	1.00	0.98	0.02
RLCR	0.45	0.39	0.06
RLSR	**0.89**	**0.52**	**0.37**

Discussion. To understand where RLSR's performance gains originate, we examine the detailed behavior of the risk-coverage curves in Figs. 5 and 6. A revealing pattern emerges: at full coverage, RLSR typically achieves comparable (or even slightly higher) risk than RLVR and RLCR, indicating similar overall accuracy. However, as coverage decreases, the RLSR's risk drops substantially faster, yielding a lower AURC. This is expected: the risk at full coverage is the population risk by definition, whereas RLSR targets AURC rather than the population risk. The gains are not limited to very low coverage: RLSR achieves the best Acc@50 in all four settings of Tab. 1. This behavior directly confirms the benefit of our alignment objective: by

optimizing a lifted AURC reward, RLSR learns to assign higher confidence to correct predictions and lower confidence to incorrect ones (see Tab. 2 for empirical separation), precisely the ranking needed for effective selection. In contrast, RLCR shows a more gradual decline, and RLVR's curve often remains flat or even increases at low coverage, implying inconsistent confidence assignment for totally ignoring confidence in RLVR.

Ablation study. Since RLSR approximates the global ranking through batch ranking, we study the effect of the ranking pool size $B \times G$ on SP performance. Fixing $G = 32$ and training Qwen2.5-7B on HotPotQA with $B = 48, 32, 16$, the ID AURC is $0.44, 0.44, 0.45$ and the OOD AURC is $0.41, 0.47, 0.48$, respectively. The ID performance is stable, while the OOD performance degrades with smaller batches. The latter degradation is concentrated on the two math OOD benchmarks (Sec. D.5). Halving the group size to $G = 16$, which halves the rollout cost, degrades AURC only slightly (ID 0.46, OOD 0.42 at $B = 48$), still outperforming RLVR and RLCR at the full cost.

4.2 Benchmarking in Risk-Controlled Deployment

To show the effectiveness of RLSR in high-stakes deployment, we apply the models trained on HotPotQA to MedQA [36], a medical question-answering benchmark derived from various professional medical board and licensing examinations. This represents a challenging OOD setting where risk control is critical.

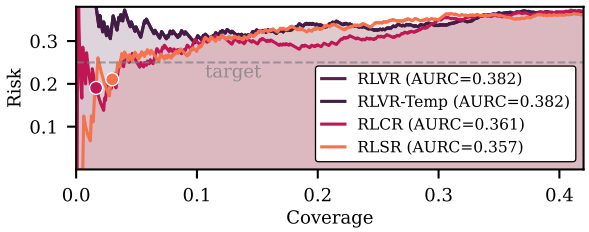


Figure 7: The risk-coverage curves of the three alignment methods on the MedQA validation set. The dashed line indicates the 25% target risk (i.e., 75% accuracy), and the dots mark the actual operating point of each method—not exactly on 25% risk level due to threshold selection on the finite validation set.

Experimental setup. Since all alignment methods on MedQA achieve full-coverage accuracy between 57% and 59% (see Fig. 7), we set a target accuracy at 75%, much higher than the full-coverage accuracy. We focus on this low-coverage, low-risk regime with a controlled risk level to emulate practical high-stakes deployment scenarios. To achieve the target accuracy, we find the cutoff threshold τ for each method on a validation set with samples $1,272$, and then apply the threshold to the test set with samples $1,273$.

Results and discussion. As shown in Tab. 3 and Fig. 7, RLVR fails to achieve the target accuracy 75% throughout the coverage range, due to their overly high and ill-calibrated confidence regardless of the prediction correctness—the very issue that RLCR aims to address [11, 16]. Post-hoc temperature scaling [5, 37] (see Sec. C.6.5) over RLVR (RLVR-Temp) does not improve or even change the SP performance, as it simply performs monotonic rescaling of the confidence score and hence preserves confidence ranking. This underscores the importance of optimizing the confidence scorers in RLCR and RLSR. RLSR's advantage is most pronounced in the low-coverage, low-risk regime (see Fig. 7): RLSR's risk-coverage curve is not only almost uniformly lower than that of RLCR's, but reaches very low risk—RLCR's reaches a significant risk floor ($\sim 15\%$) first and then bounces back as coverage level decreases. For error control (see Tab. 3), only RLSR and RLCR achieve the 75% target accuracy on the validation set but have very different generalization behaviors: while RLCR's test accuracy drops below the target (81.0% validation $\rightarrow$ 72.2% test), RLSR has the same test accuracy and coverage as the validation ones. Overall, RLSR obtains simultaneous higher accuracy and coverage than RLCR's by large margins on the test set, demonstrating RLSR's strong error-controlled SP performance. The absolute coverage of all methods for this stringent target is low. Whether it is acceptable depends on the cost of errors relative to that of abstention in the application. For a fixed target risk, absolute coverage

Table 3: SP performance of the three methods on MedQA (target accuracy: 75%). Temperature scaling (`RLVR-Temp`) compresses RLVR's confidence range to $[0.38, 0.62]$, so $\tau = 0.62$ is the maximum achievable threshold; $(\cdot)^\dagger$ highlights methods not able to achieve the target accuracy.

Method	τ	Val. Cov.	Val. Acc.	Test Cov.	Test Acc.
RLVR†	1.0	–	66.6%	–	–
RLVR-Temp†	0.62	–	66.6%	–	–
RLCR	0.95	1.7%	81.0%	1.4%	72.2%
RLSR	1.0	3.0%	78.9%	3.0%	**78.9%**

can be increased with a stronger LLM and confidence scorer, or with a risk-controlled alignment objective (Sec. 5).

5 Conclusion

For an LLM to be reliable in high-stakes deployment, it must not only answer correctly but also know when to answer. Our central message is that calibration is neither sufficient nor necessary for this goal: selective prediction (SP) depends on whether correct answers are ranked ahead of incorrect ones by confidence, which calibration does not guarantee. Hence, if the goal is SP, LLMs should be aligned with SP directly. Our alignment method, RLSR, does so by optimizing the area under the risk-coverage curve (AURC) as a reward. In our evaluation across in-domain and out-of-domain benchmarks, RLSR achieves significantly better risk-coverage tradeoff than accuracy-oriented (RLVR) and calibration-oriented (RLCR) alignment; and in a risk-controlled simulated medical deployment task, it is the only method that meets the target accuracy with reasonable coverage.

Limitations and Future Works and Broader Impacts

Limitations. **First**, AURC summarizes the entire risk-coverage tradeoff, so optimizing it does not guarantee better coverage at any target risk level, and coverage at a stringent risk could be very low (Sec. 4.2). **Second**, RLSR optimizes the confidence ranking rather than its calibration, so its ECE is higher than that of RLCR. This reflects a practical tension between calibration and selection. In fact, the only possibility they are perfectly aligned is when all correct predictions get confidence 1 and all incorrect confidence 0—unlikely attainable in practice. **Third**, RLSR approximates the population-level ranking in AURC by ranking within a batch, which introduces finite-sample approximation noise. The OOD performance degrades with smaller batches (Sec. 4.1), and we do not provide a theoretical characterization of this noise. **Fourth**, like other alignment methods that rely on a verifier, RLSR requires automatically verifiable correctness, so open-ended generation and multi-turn dialog remain outside its current scope.

Future works. When the target risk is known, one could formulate risk-controlled coverage maximization as the alignment objective. Beyond global calibration, which is misaligned with SP, one could pursue calibration on the selected subset [38]. A finite-sample analysis of batch ranking would clarify how batch size affects the approximation of AURC. Extending RLSR to open-ended and multi-turn tasks would be extremely interesting.

Broader Impacts. This work focuses on the refusal capacity of LLMs so that the retained subset can be more accurate. The potential positive societal impacts include reliable decision making in high-stakes domains because the users can expect that the returned answer comes from the retained subset, and thus the predictive accuracy is close to the target accuracy set for pre-deployment calibration. Just like any optimization methods, if users turn the risk axis in AURC into any misuse metric, this technology can be turned into a selective prediction system for selectively returning the misuse information only. At the end of the day, it is really up to the intent of the users.

Acknowledgments and Disclosure of Funding

This work was supported in part by a UMN DSI-MnDRIVE PhD Graduate Assistantship, the National Institutes of Health (NIH) under award No. R01CA287413, R01NS131314, R01MH138929, U01FD008720, the National Science Foundation (NSF) under award No. 2435911 and 2541023. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. The authors acknowledge the Minnesota Supercomputing Institute (MSI) at the University of Minnesota for providing resources that contributed to the research results reported within this paper.

References

[1] J. Lai, W. Gan, J. Wu, Z. Qi, and P. S. Yu, "Large language models in law: A survey," *AI Open*, vol. 5, pp. 181–196, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666651024000172

[2] T. Fan, Y. Yang, Y. Jiang, Y. Zhang, Y. Chen, and C. Huang, "Ai-trader: Benchmarking autonomous agents in real-time financial markets," 2025. [Online]. Available: https://arxiv.org/abs/2512.10971

[3] F. Gaber, M. Shaik, F. Allega, A. J. Bilecz, F. Busch, K. Goon, V. Franke, and A. Akalin, "Evaluating large language model workflows in clinical decision support for triage and referral and diagnosis," *npj Digital Medicine*, vol. 8, no. 1, p. 263, May 2025. [Online]. Available: https://doi.org/10.1038/s41746-025-01684-1

[4] A. T. Kalai, O. Nachum, S. S. Vempala, and E. Zhang, "Evaluating large language models for accuracy incentivizes hallucinations," *Nature*, Apr 2026. [Online]. Available: https://doi.org/10.1038/s41586-026-10549-w

[5] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International conference on machine learning*. PMLR, 2017, pp. 1321–1330.

[6] J. Nixon, M. W. Dusenberry, L. Zhang, G. Jerfel, and D. Tran, "Measuring calibration in deep learning." in *CVPR workshops*, vol. 2, 2019.

[7] A. Fisch, T. S. Jaakkola, and R. Barzilay, "Calibrated selective classification," *Transactions on Machine Learning Research*, 2022.

[8] H. Liang, L. Peng, and J. Sun, "Selective classification under distribution shifts," *Transactions on Machine Learning Research*, 2024. [Online]. Available: https://openreview.net/forum?id=dmxMGW6J7N

[9] S. Zhang, C. Chen, G. Li, and J. Sun, "Ensemble selective classification," in *The Fortieth Annual Conference on Neural Information Processing Systems*, 2026. [Online]. Available: https://openreview.net/forum?id=Qqq3JkX68b

[10] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, M. Zhang, Y. K. Li, Y. Wu, and D. Guo, "Deepseekmath: Pushing the limits of mathematical reasoning in open language models," *CoRR*, vol. abs/2402.03300, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2402.03300

[11] M. Damani, I. Puri, S. Slocum, I. Shenfeld, L. Choshen, Y. Kim, and J. Andreas, "Beyond binary rewards: Training LMs to reason about their uncertainty," in *The Fourteenth International Conference on Learning Representations*, 2026. [Online]. Available: https://openreview.net/forum?id=ASQ649zdHm

[12] D. Guo, D. Yang, H. Zhang, J. Song, P. Wang, Q. Zhu, R. Xu, R. Zhang, S. Ma, X. Bi, X. Zhang, X. Yu, Y. Wu, Z. F. Wu, Z. Gou, Z. Shao, Z. Li, Z. Gao, A. Liu, B. Xue, B. Wang, B. Wu, B. Feng, C. Lu, C. Zhao, C. Deng, C. Ruan, D. Dai, D. Chen, D. Ji, E. Li, F. Lin, F. Dai, F. Luo, G. Hao, G. Chen, G. Li, H. Zhang, H. Xu, H. Ding, H. Gao, H. Qu, H. Li, J. Guo, J. Li, J. Chen, J. Yuan, J. Tu, J. Qiu, J. Li, J. L. Cai, J. Ni, J. Liang, J. Chen, K. Dong, K. Hu, K. You, K. Gao, K. Guan, K. Huang, K. Yu, L. Wang, L. Zhang, L. Zhao, L. Wang, L. Zhang, L. Xu, L. Xia, M. Zhang, M. Zhang, M. Tang, M. Zhou, M. Li, M. Wang, M. Li, N. Tian, P. Huang, P. Zhang, Q. Wang, Q. Chen, Q. Du, R. Ge, R. Zhang, R. Pan, R. Wang, R. J. Chen, R. L. Jin, R. Chen, S. Lu, S. Zhou, S. Chen, S. Ye, S. Wang, S. Yu, S. Zhou, S. Pan, S. S. Li, S. Zhou, S. Wu, T. Yun, T. Pei, T. Sun, T. Wang, W. Zeng, W. Liu, W. Liang, W. Gao, W. Yu, W. Zhang, W. L. Xiao, W. An, X. Liu, X. Wang, X. Chen, X. Nie, X. Cheng, X. Liu, X. Xie, X. Liu, X. Yang, X. Li, X. Su, X. Lin, X. Q. Li, X. Jin, X. Shen, X. Chen, X. Sun, X. Wang, X. Song, X. Zhou, X. Wang, X. Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. Zhang, Y. Xu, Y. Li, Y. Zhao, Y. Sun, Y. Wang, Y. Yu, Y. Zhang, Y. Shi, Y. Xiong, Y. He, Y. Piao, Y. Wang, Y. Tan, Y. Ma, Y. Liu, Y. Guo, Y. Ou, Y. Wang, Y. Gong, Y. Zou, Y. He, Y. Xiong, Y. Luo, Y. You, Y. Liu, Y. Zhou, Y. X. Zhu, Y. Huang, Y. Li, Y. Zheng, Y. Zhu, Y. Ma, Y. Tang, Y. Zha, Y. Yan, Z. Z. Ren, Z. Ren, Z. Sha, Z. Fu, Z. Xu, Z. Xie, Z. Zhang, Z. Hao, Z. Ma, Z. Yan, Z. Wu, Z. Gu, Z. Zhu, Z. Liu, Z. Li, Z. Xie, Z. Song, Z. Pan, Z. Huang, Z. Xu, Z. Zhang, and Z. Zhang, "Deepseek-r1 incentivizes reasoning in llms through reinforcement learning," *Nature*, vol. 645, no. 8081, pp. 633–638, Sep 2025. [Online]. Available: https://doi.org/10.1038/s41586-025-09422-z

[13] X. Wen, Z. Liu, S. Zheng, S. Ye, Z. Wu, Y. Wang, Z. Xu, X. Liang, J. Li, Z. Miao, J. Bian, and M. Yang, "Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base LLMs," in *The Fourteenth International Conference on Learning Representations*, 2026. [Online]. Available: https://openreview.net/forum?id=jGbRWwIidy

[14] D. Kumaran, A. Conmy, F. Barbero, S. Osindero, V. Patraucean, and P. Veličković, "How do LLMs compute verbal confidence?" in *Forty-third International Conference on Machine Learning*, 2026. [Online]. Available: https://openreview.net/forum?id=t7oQT3uX66

[15] M. Xiong, Z. Hu, X. Lu, Y. LI, J. Fu, J. He, and B. Hooi, "Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=gjeQKFxFpZ

[16] J. Leng, C. Huang, B. Zhu, and J. Huang, "Taming overconfidence in LLMs: Reward calibration in RLHF," in *The Thirteenth International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=l0tg0jzsdL

[17] H. Zhou, J. V. Landeghem, T. Popordanoska, and M. B. Blaschko, "A novel characterization of the population area under the risk coverage curve (AURC) and rates of finite sample estimators," in *Forty-second International Conference on Machine Learning*, 2025. [Online]. Available: https://openreview.net/forum?id=LBBUJkqkOM

[18] J. Geng, F. Cai, Y. Wang, H. Koeppl, P. Nakov, and I. Gurevych, "A survey of confidence estimation and calibration in large language models," in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, K. Duh, H. Gomez, and S. Bethard, Eds. Mexico City, Mexico: Association for Computational Linguistics, Jun. 2024, pp. 6577–6595. [Online]. Available: https://aclanthology.org/2024.naacl-long.366/

[19] Z. Xia, J. Xu, Y. Zhang, and H. Liu, "A survey of uncertainty estimation methods on large language models," in *Findings of the Association for Computational Linguistics: ACL 2025*, W. Che, J. Nabende, E. Shutova, and M. T. Pilehvar, Eds. Vienna, Austria: Association for Computational Linguistics, Jul. 2025, pp. 21 381–21 396. [Online]. Available: https://aclanthology.org/2025.findings-acl.1101/

[20] S. Rabanser and N. Papernot, "What does it take to build a performant selective classifier?" in *Advances in Neural Information Processing Systems*, 2025.

[21] J. Chen, J. Yoon, S. Ebrahimi, S. O. Arik, T. Pfister, and S. Jha, "Adaptation with self-evaluation to improve selective prediction in LLMs," in *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. [Online]. Available: https://openreview.net/forum?id=haPIkA8aOk

[22] Y. Mao, T. Durand, N. Mehrasa, J. He, and M. Ester, "Calibrating LLMs for selective prediction: Balancing coverage and risk," in *Socially Responsible and Trustworthy Foundation Models at NeurIPS 2025*, 2025. [Online]. Available: https://openreview.net/forum?id=ZVZGjtP5VB

[23] V. Franc, D. Prusa, and V. Voracek, "Optimal strategies for reject option classifiers," *Journal of Machine Learning Research*, vol. 24, no. 11, pp. 1–49, 2023. [Online]. Available: http://jmlr.org/papers/v24/21-0048.html

[24] Qwen, :, A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Tang, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, and Z. Qiu, "Qwen2.5 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2412.15115

[25] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, A. Yang, A. Fan, A. Goyal, A. Hartshorn, A. Yang, A. Mitra, A. Sravankumar, A. Korenev, A. Hinsvark, A. Rao, A. Zhang, A. Rodriguez, A. Gregerson, A. Spataru, B. Roziere, B. Biron, B. Tang, B. Chern, C. Caucheteux, C. Nayak, C. Bi, C. Marra, C. McConnell, C. Keller, C. Touret, C. Wu, C. Wong, C. C. Ferrer, C. Nikolaidis, D. Allonsius, D. Song, D. Pintz, D. Livshits, D. Wyatt, D. Esiobu, D. Choudhary, D. Mahajan, D. Garcia-Olano, D. Perino, D. Hupkes, E. Lakomkin, E. AlBadawy, E. Lobanova, E. Dinan, E. M. Smith, F. Radenovic, F. Guzmán, F. Zhang, G. Synnaeve, G. Lee, G. L. Anderson, G. Thattai, G. Nail, G. Mialon, G. Pang, G. Cucurell, H. Nguyen, H. Korevaar, H. Xu,

H. Touvron, I. Zarov, I. A. Ibarra, I. Kloumann, I. Misra, I. Evtimov, J. Zhang, J. Copet, J. Lee, J. Geffert, J. Vranes, J. Park, J. Mahadeokar, J. Shah, J. van der Linde, J. Billock, J. Hong, J. Lee, J. Fu, J. Chi, J. Huang, J. Liu, J. Wang, J. Yu, J. Bitton, J. Spisak, J. Park, J. Rocca, J. Johnstun, J. Saxe, J. Jia, K. V. Alwala, K. Prasad, K. Upasani, K. Plawiak, K. Li, K. Heafield, K. Stone, K. El-Arini, K. Iyer, K. Malik, K. Chiu, K. Bhalla, K. Lakhotia, L. Rantala-Yeary, L. van der Maaten, L. Chen, L. Tan, L. Jenkins, L. Martin, L. Madaan, L. Malo, L. Blecher, L. Landzaat, L. de Oliveira, M. Muzzi, M. Pasupuleti, M. Singh, M. Paluri, M. Kardas, M. Tsimpoukelli, M. Oldham, M. Rita, M. Pavlova, M. Kambadur, M. Lewis, M. Si, M. K. Singh, M. Hassan, N. Goyal, N. Torabi, N. Bashlykov, N. Bogoychev, N. Chatterji, N. Zhang, O. Duchenne, O. Çelebi, P. Alrassy, P. Zhang, P. Li, P. Vasic, P. Weng, P. Bhargava, P. Dubal, P. Krishnan, P. S. Koura, P. Xu, Q. He, Q. Dong, R. Srinivasan, R. Ganapathy, R. Calderer, R. S. Cabral, R. Stojnic, R. Raileanu, R. Maheswari, R. Girdhar, R. Patel, R. Sauvestre, R. Polidoro, R. Sumbaly, R. Taylor, R. Silva, R. Hou, R. Wang, S. Hosseini, S. Chennabasappa, S. Singh, S. Bell, S. S. Kim, S. Edunov, S. Nie, S. Narang, S. Raparthy, S. Shen, S. Wan, S. Bhosale, S. Zhang, S. Vandenhende, S. Batra, S. Whitman, S. Sootla, S. Collot, S. Gururangan, S. Borodinsky, T. Herman, T. Fowler, T. Sheasha, T. Georgiou, T. Scialom, T. Speckbacher, T. Mihaylov, T. Xiao, U. Karn, V. Goswami, V. Gupta, V. Ramanathan, V. Kerkez, V. Gonguet, V. Do, V. Vogeti, V. Albiero, V. Petrovic, W. Chu, W. Xiong, W. Fu, W. Meers, X. Martinet, X. Wang, X. Wang, X. E. Tan, X. Xia, X. Xie, X. Jia, X. Wang, Y. Goldschlag, Y. Gaur, Y. Babaei, Y. Wen, Y. Song, Y. Zhang, Y. Li, Y. Mao, Z. D. Coudert, Z. Yan, Z. Chen, Z. Papakipos, A. Singh, A. Srivastava, A. Jain, A. Kelsey, A. Shajnfeld, A. Gangidi, A. Victoria, A. Goldstand, A. Menon, A. Sharma, A. Boesenberg, A. Baevski, A. Feinstein, A. Kallet, A. Sangani, A. Teo, A. Yunus, A. Lupu, A. Alvarado, A. Caples, A. Gu, A. Ho, A. Poulton, A. Ryan, A. Ramchandani, A. Dong, A. Franco, A. Goyal, A. Saraf, A. Chowdhury, A. Gabriel, A. Bharambe, A. Eisenman, A. Yazdan, B. James, B. Maurer, B. Leonhardi, B. Huang, B. Loyd, B. D. Paola, B. Paranjape, B. Liu, B. Wu, B. Ni, B. Hancock, B. Wasti, B. Spence, B. Stojkovic, B. Gamido, B. Montalvo, C. Parker, C. Burton, C. Mejia, C. Liu, C. Wang, C. Kim, C. Zhou, C. Hu, C.-H. Chu, C. Cai, C. Tindal, C. Feichtenhofer, C. Gao, D. Civin, D. Beaty, D. Kreymer, D. Li, D. Adkins, D. Xu, D. Testuggine, D. David, D. Parikh, D. Liskovich, D. Foss, D. Wang, D. Le, D. Holland, E. Dowling, E. Jamil, E. Montgomery, E. Presani, E. Hahn, E. Wood, E.-T. Le, E. Brinkman, E. Arcaute, E. Dunbar, E. Smothers, F. Sun, F. Kreuk, F. Tian, F. Kokkinos, F. Ozgenel, F. Caggioni, F. Kanayet, F. Seide, G. M. Florez, G. Schwarz, G. Badeer, G. Swee, G. Halpern, G. Herman, G. Sizov, Guangyi, Zhang, G. Lakshminarayanan, H. Inan, H. Shojanazeri, H. Zou, H. Wang, H. Zha, H. Habeeb, H. Rudolph, H. Suk, H. Aspegren, H. Goldman, H. Zhan, I. Damlaj, I. Molybog, I. Tufanov, I. Leontiadis, I.-E. Veliche, I. Gat, J. Weissman, J. Geboski, J. Kohli, J. Lam, J. Asher, J.-B. Gaya, J. Marcus, J. Tang, J. Chan, J. Zhen, J. Reizenstein, J. Teboul, J. Zhong, J. Jin, J. Yang, J. Cummings, J. Carvill, J. Shepard, J. McPhie, J. Torres, J. Ginsburg, J. Wang, K. Wu, K. H. U, K. Saxena, K. Khandelwal, K. Zand, K. Matosich, K. Veeraraghavan, K. Michelena, K. Li, K. Jagadeesh, K. Huang, K. Chawla, K. Huang, L. Chen, L. Garg, L. A, L. Silva, L. Bell, L. Zhang, L. Guo, L. Yu, L. Moshkovich, L. Wehrstedt, M. Khabsa, M. Avalani, M. Bhatt, M. Mankus, M. Hasson, M. Lennie, M. Reso, M. Groshev, M. Naumov, M. Lathi, M. Keneally, M. Liu, M. L. Seltzer, M. Valko, M. Restrepo, M. Patel, M. Vyatskov, M. Samvelyan, M. Clark, M. Macey, M. Wang, M. J. Hermoso, M. Metanat, M. Rastegari, M. Bansal, N. Santhanam, N. Parks, N. White, N. Bawa, N. Singhal, N. Egebo, N. Usunier, N. Mehta, N. P. Laptev, N. Dong, N. Cheng, O. Chernoguz, O. Hart, O. Salpekar, O. Kalinli, P. Kent, P. Parekh, P. Saab, P. Balaji, P. Rittner, P. Bontrager, P. Roux, P. Dollar, P. Zvyagina, P. Ratanchandani, P. Yuvraj, Q. Liang, R. Alao, R. Rodriguez, R. Ayub, R. Murthy, R. Nayani, R. Mitra, R. Parthasarathy, R. Li, R. Hogan, R. Battey, R. Wang, R. Howes, R. Rinott, S. Mehta, S. Siby, S. J. Bondu, S. Datta, S. Chugh, S. Hunt, S. Dhillon, S. Sidorov, S. Pan, S. Mahajan, S. Verma, S. Yamamoto, S. Ramaswamy, S. Lindsay, S. Lindsay, S. Feng, S. Lin, S. C. Zha, S. Patil, S. Shankar, S. Zhang, S. Zhang, S. Wang, S. Agarwal, S. Sajuyigbe, S. Chintala, S. Max, S. Chen, S. Kehoe, S. Satterfield, S. Govindaprasad, S. Gupta, S. Deng, S. Cho, S. Virk, S. Subramanian, S. Choudhury, S. Goldman, T. Remez, T. Glaser, T. Best, T. Koehler, T. Robinson, T. Li, T. Zhang, T. Matthews, T. Chou, T. Shaked, V. Vontimitta, V. Ajayi, V. Montanez, V. Mohan, V. S. Kumar, V. Mangla, V. Ionescu, V. Poenaru, V. T. Mihailescu, V. Ivanov, W. Li, W. Wang, W. Jiang, W. Bouaziz, W. Constable, X. Tang, X. Wu, X. Wang, X. Wu, X. Gao, Y. Kleinman, Y. Chen, Y. Hu, Y. Jia, Y. Qi, Y. Li, Y. Zhang, Y. Zhang, Y. Adi, Y. Nam, Yu, Wang, Y. Zhao, Y. Hao, Y. Qian, Y. Li, Y. He, Z. Rait, Z. DeVito, Z. Rosnbrick, Z. Wen, Z. Yang, Z. Zhao, and Z. Ma, "The llama 3 herd of models,"

2024. [Online]. Available: https://arxiv.org/abs/2407.21783

[26] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. D. Manning, "HotpotQA: A dataset for diverse, explainable multi-hop question answering," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, Eds. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 2369–2380. [Online]. Available: https://aclanthology.org/D18-1259/

[27] A. Albalak, D. Phung, N. Lile, R. Rafailov, K. Gandhi, L. Castricato, A. Singh, C. Blagden, V. Xiang, D. Mahan, and N. Haber, "Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models," 2025. [Online]. Available: https://arxiv.org/abs/2502.17387

[28] J. Wei, N. Karina, H. W. Chung, Y. J. Jiao, S. Papay, A. Glaese, J. Schulman, and W. Fedus, "Measuring short-form factuality in large language models," 2024. [Online]. Available: https://arxiv.org/abs/2411.04368

[29] M. Joshi, E. Choi, D. Weld, and L. Zettlemoyer, "TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, R. Barzilay and M.-Y. Kan, Eds. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 1601–1611. [Online]. Available: https://aclanthology.org/P17-1147/

[30] D. Rein, B. L. Hou, A. C. Stickland, J. Petty, R. Y. Pang, J. Dirani, J. Michael, and S. R. Bowman, "GPQA: A graduate-level google-proof q&a benchmark," in *First Conference on Language Modeling*, 2024. [Online]. Available: https://openreview.net/forum?id=Ti67584b98

[31] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt, "Measuring mathematical problem solving with the MATH dataset," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. [Online]. Available: https://openreview.net/forum?id=7Bywt2mQsCe

[32] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman, "Training verifiers to solve math word problems," 2021. [Online]. Available: https://arxiv.org/abs/2110.14168

[33] A. Talmor, J. Herzig, N. Lourie, and J. Berant, "CommonsenseQA: A question answering challenge targeting commonsense knowledge," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4149–4158. [Online]. Available: https://aclanthology.org/N19-1421/

[34] B. Turtel, D. Franklin, K. Skotheim, L. Hewitt, and P. Schoenegger, "Outcome-based reinforcement learning to predict the future," *Transactions on Machine Learning Research*, 2025. [Online]. Available: https://openreview.net/forum?id=bbhdeL8EUX

[35] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le, and C. Sutton, "Program synthesis with large language models," 2021. [Online]. Available: https://arxiv.org/abs/2108.07732

[36] D. Jin, E. Pan, N. Oufattole, W.-H. Weng, H. Fang, and P. Szolovits, "What disease does this patient have? a large-scale open domain question answering dataset from medical exams," *Applied Sciences*, vol. 11, no. 14, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/14/6421

[37] C. Wang, G. Szarvas, G. Balazs, P. Danchenko, and P. Ernst, "Calibrating verbalized probabilities for large language models," 2024. [Online]. Available: https://arxiv.org/abs/2410.06707

[38] A. Fisch, T. S. Jaakkola, and R. Barzilay, "Calibrated selective classification," *Transactions on Machine Learning Research*, 2022. [Online]. Available: https://openreview.net/forum?id=zFhNBs8GaV

[39] J. Cole, M. Zhang, D. Gillick, J. Eisenschlos, B. Dhingra, and J. Eisenstein, "Selectively answering ambiguous questions," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, H. Bouamor, J. Pino, and K. Bali, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 530–543. [Online]. Available: https://aclanthology.org/2023.emnlp-main.35/

[40] N. Varshney and C. Baral, "Post-abstention: Towards reliably re-attempting the abstained instances in QA," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, A. Rogers, J. Boyd-Graber, and N. Okazaki, Eds. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 967–982. [Online]. Available: https://aclanthology.org/2023.acl-long.55/

[41] V. Quach, A. Fisch, T. Schuster, A. Yala, J. H. Sohn, T. S. Jaakkola, and R. Barzilay, "Conformal language modeling," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=pzUhfQ74c5

[42] X. Liang, H. Wang, Y. Wang, S. Song, J. Yang, S. Niu, J. Hu, D. Liu, S. Yao, F. Xiong *et al.*, "Controllable text generation for large language models: A survey," *arXiv preprint arXiv:2408.12599*, 2024.

[43] K. Tian, E. Mitchell, A. Zhou, A. Sharma, R. Rafailov, H. Yao, C. Finn, and C. Manning, "Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, H. Bouamor, J. Pino, and K. Bali, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 5433–5442. [Online]. Available: https://aclanthology.org/2023.emnlp-main.330/

[44] T. A. Lamb, D. R. Ivanova, P. Torr, and T. G. J. Rudner, "Semantic calibration of LLMs through the lens of temperature scaling," in *ICLR Workshop: Quantify Uncertainty and Hallucination in Foundation Models: The Next Frontier in Reliable AI*, 2025. [Online]. Available: https://openreview.net/forum?id=22UNXHzBgt

[45] M. Zhang, M. Huang, R. Shi, L. Guo, C. Peng, P. Yan, Y. Zhou, and X. Qiu, "Calibrating the confidence of large language models by eliciting fidelity," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, Eds. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 2959–2979. [Online]. Available: https://aclanthology.org/2024.emnlp-main.173/

[46] G. Luo and A. Deshwal, "COM-BOM: Bayesian exemplar search for efficiently exploring the accuracy-calibration Pareto frontier," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, Eds. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 20 350–20 363. [Online]. Available: https://aclanthology.org/2025.emnlp-main.1027/

[47] Q. Lyu, K. Shridhar, C. Malaviya, L. Zhang, Y. Elazar, N. Tandon, M. Apidianaki, M. Sachan, and C. Callison-Burch, "Calibrating large language models with sample consistency," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 18, pp. 19 260–19 268, Apr. 2025. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/34120

[48] J. Hu, Y. Zhang, Q. Han, D. Jiang, X. Zhang, and H.-Y. Shum, "Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. [Online]. Available: https://openreview.net/forum?id=NFM8F5cV0V

[49] C. Xiao, M. Zhang, and Y. Cao, "BNPO: Beta normalization policy optimization," in *Findings of the Association for Computational Linguistics: EMNLP 2026*, H.-Y. Lee, S. Dev, J. Li, and D. Elliott, Eds. Budapest, Hungary: Association for Computational Linguistics, Oct. 2026, pp. 50 817–50 829. [Online]. Available: https://aclanthology.org/2026.findings-emnlp.2364/

[50] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=nZeVKeeFYf9

[51] J. Schulman and T. M. Lab, "Lora without regret," *Thinking Machines Lab: Connectionism*, 2025, https://thinkingmachines.ai/blog/lora/.

[52] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica, "Efficient memory management for large language model serving with pagedattention," in *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

[53] T. Dao, "FlashAttention-2: Faster attention with better parallelism and work partitioning," in *International Conference on Learning Representations (ICLR)*, 2024.

[54] K. Saab, T. Tu, W.-H. Weng, R. Tanno, D. Stutz, E. Wulczyn, F. Zhang, T. Strother, C. Park, E. Vedadi, J. Z. Chaves, S.-Y. Hu, M. Schaekermann, A. Kamath, Y. Cheng, D. G. T. Barrett, C. Cheung, B. Mustafa, A. Palepu, D. McDuff, L. Hou, T. Golany, L. Liu, J. baptiste Alayrac, N. Houlsby, N. Tomasev, J. Freyberg, C. Lau, J. Kemp, J. Lai, S. Azizi, K. Kanada, S. Man, K. Kulkarni, R. Sun, S. Shakeri, L. He, B. Caine, A. Webson, N. Latysheva, M. Johnson, P. Mansfield, J. Lu, E. Rivlin, J. Anderson, B. Green, R. Wong, J. Krause, J. Shlens, E. Dominowska, S. M. A. Eslami, K. Chou, C. Cui, O. Vinyals, K. Kavukcuoglu, J. Manyika, J. Dean, D. Hassabis, Y. Matias, D. Webster, J. Barral, G. Corrado, C. Semturs, S. S. Mahdavi, J. Gottweis, A. Karthikesalingam, and V. Natarajan, "Capabilities of gemini models in medicine," 2024. [Online]. Available: https://arxiv.org/abs/2404.18416

[55] A. Sellergren, S. Kazemzadeh, T. Jaroensri, A. Kiraly, M. Traverse, T. Kohlberger, S. Xu, F. Jamil, C. Hughes, C. Lau, J. Chen, F. Mahvar, L. Yatziv, T. Chen, B. Sterling, S. A. Baby, S. M. Baby, J. Lai, S. Schmidgall, L. Yang, K. Chen, P. Bjornsson, S. Reddy, R. Brush, K. Philbrick, M. Asiedu, I. Mezerreg, H. Hu, H. Yang, R. Tiwari, S. Jansen, P. Singh, Y. Liu, S. Azizi, A. Kamath, J. Ferret, S. Pathak, N. Vieillard, R. Merhej, S. Perrin, T. Matejovicova, A. Ramé, M. Riviere, L. Rouillard, T. Mesnard, G. Cideron, J. bastien Grill, S. Ramos, E. Yvinec, M. Casbon, E. Buchatskaya, J.-B. Alayrac, D. Lepikhin, V. Feinberg, S. Borgeaud, A. Andreev, C. Hardin, R. Dadashi, L. Hussenot, A. Joulin, O. Bachem, Y. Matias, K. Chou, A. Hassidim, K. Goel, C. Farabet, J. Barral, T. Warkentin, J. Shlens, D. Fleet, V. Cotruta, O. Sanseviero, G. Martins, P. Kirk, A. Rao, S. Shetty, D. F. Steiner, C. Kirmizibayrak, R. Pilgrim, D. Golden, and L. Yang, "Medgemma technical report," 2025. [Online]. Available: https://arxiv.org/abs/2507.05201

[56] OpenAI, S. Agarwal, L. Ahmad, J. Ai, S. Altman, A. Applebaum, E. Arbus, R. K. Arora, Y. Bai, B. Baker, H. Bao, B. Barak, A. Bennett, T. Bertao, N. Brett, E. Brevdo, G. Brockman, S. Bubeck, C. Chang, K. Chen, M. Chen, E. Cheung, A. Clark, D. Cook, M. Dukhan, C. Dvorak, K. Fives, V. Fomenko, T. Garipov, K. Georgiev, M. Glaese, T. Gogineni, A. Goucher, L. Gross, K. G. Guzman, J. Hallman, J. Hehir, J. Heidecke, A. Helyar, H. Hu, R. Huet, J. Huh, S. Jain, Z. Johnson, C. Koch, I. Kofman, D. Kundel, J. Kwon, V. Kyrylov, E. Y. Le, G. Leclerc, J. P. Lennon, S. Lessans, M. Lezcano-Casado, Y. Li, Z. Li, J. Lin, J. Liss, Lily, Liu, J. Liu, K. Lu, C. Lu, Z. Martinovic, L. McCallum, J. McGrath, S. McKinney, A. McLaughlin, S. Mei, S. Mostovoy, T. Mu, G. Myles, A. Neitz, A. Nichol, J. Pachocki, A. Paino, D. Palmie, A. Pantuliano, G. Parascandolo, J. Park, L. Pathak, C. Paz, L. Peran, D. Pimenov, M. Pokrass, E. Proehl, H. Qiu, G. Raila, F. Raso, H. Ren, K. Richardson, D. Robinson, B. Rotsted, H. Salman, S. Sanjeev, M. Schwarzer, D. Sculley, H. Sikchi, K. Simon, K. Singhal, Y. Song, D. Stuckey, Z. Sun, P. Tillet, S. Toizer, F. Tsimpourlas, N. Vyas, E. Wallace, X. Wang, M. Wang, O. Watkins, K. Weil, A. Wendling, K. Whinnery, C. Whitney, H. Wong, L. Yang, Y. Yang, M. Yasunaga, K. Ying, W. Zaremba, W. Zhan, C. Zhang, B. Zhang, E. Zhang, and S. Zhao, "gpt-oss-120b & gpt-oss-20b model card," 2025. [Online]. Available: https://arxiv.org/abs/2508.10925

A Derivations & Proofs

A.1 Gradient of the LLM alignment objective in Eq. (2.1)

First of all,

$$\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}} \mathbb{E}_{\widehat{y}\sim\pi_\theta(\cdot|x)} R(y,\widehat{y}) = \mathbb{E}_{(x,y)\sim\mathcal{D}} \nabla_\theta \mathbb{E}_{\widehat{y}\sim\pi_\theta(\cdot|x)} R(y,\widehat{y}), \tag{A.1}$$

so we can focus on deriving $\nabla_\theta \mathbb{E}_{\widehat{y}\sim\pi_\theta(\cdot|x)} R(y,\widehat{y})$. Now

$$\nabla_\theta \mathbb{E}_{\widehat{y}\sim\pi_\theta(\cdot|x)} R(y,\widehat{y}) = \nabla_\theta \sum\nolimits_{\widehat{y}} \pi_\theta(\widehat{y}|x) R(y,\widehat{y}) \tag{A.2}$$

$$= \sum\nolimits_{\widehat{y}} \nabla_\theta \pi_\theta(\widehat{y}|x) R(y,\widehat{y}) \tag{A.3}$$

$$= \sum\nolimits_{\widehat{y}} \pi_\theta(\widehat{y}|x) \nabla_\theta \log \pi_\theta(\widehat{y}|x) R(y,\widehat{y}) \quad \text{(log-derivative trick)} \tag{A.4}$$

$$= \mathbb{E}_{\widehat{y}\sim\pi_\theta(\cdot|x)} [\nabla_\theta \log \pi_\theta(\widehat{y}|x) R(y,\widehat{y})], \tag{A.5}$$

as claimed.

B More Related Work

Selective algorithms for LLMs. While our paper focuses on LLM alignment and optimizing LLMs for SP, several other lines of orthogonal work focusing on other aspects of LLM deployment also contain selection components. For example, selective answering [39] deals with selecting among inherently ambiguous questions; post-abstention [40] addresses the question of what to do after abstention, i.e., refine selective prediction; conformal LLM [41] adapts conformal prediction, which outputs uncertainty sets, for LLMs; and guided/constrained decoding [42] modifies the LLM autoregressive generation process by selecting and filtering tokens to ensure its output respects prescribed constraints.

Confidence calibration (CC) for LLMs. Overconfidence in LLMs trained with RL with human feedback (RLHF) is widely recognized [11, 16]. So, CC has been a dominant line of research around it [43], evidenced by a wide range of post-hoc CC methods (e.g., temperature scaling) shown to improve calibration metrics such as expected calibration error (ECE) [44–47]. But, our focus of this paper is selective prediction, as discussed in Sec. 1.

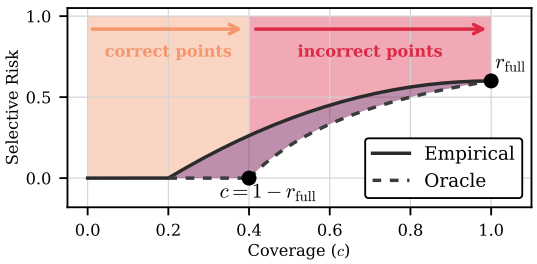


Figure 8: The dashed curve is the oracle under which coverage levels left of $c = 1 - r_{\text{full}}$ (orange) accept all correct predictions first and rank incorrect predictions last (red). r_{full} means full-coverage risk. The purple shaded area visualizes the SP gap.

Selective prediction (SP) gap. In SP, the ideal selector should mimic the *perfect-ordering oracle*, which ranks all correct predictions ahead of all incorrect ones in terms of confidence, thus minimizing the AURC. However, practical selectors typically fall short of this performance ceiling. [20] formalizes this shortfall as the *SP gap* (see Fig. 8) and decomposes it into four major sources: Bayes noise, approximation limit, statistical noise, and ranking error. For LLMs, while the approximation limit and statistical noise can be collectively reduced with larger model size and data volume during training, the major obstacle to closing the gap comes from the ranking error, which is determined by the confidence scorer. The goal of our work is to improve the LLMs capabilities as simultaneous predictors and confidence scorers by aligning them with SP, thus improving the ranking error. [20] also points out that many post-hoc calibration methods do not improve the ranking error because they are monotone transformations of the original scores and hence leave the ranking of samples intact.

C Experiment Details

C.1 Prompt Templates

We use structured prompts that guide the LLM to generate both answers and confidence scores. Following [11], we prompt the LLM to produce outputs with four components: (1) a thinking

trace enclosed in <think> tags, (2) a final answer in <answer> tags, (3) an uncertainty analysis in <analysis> tags, and (4) a numerical confidence score in <confidence> tags:

Example Response

<think> The problem is asking for the person who established the Journal of Libertarian Studies as well as [...] but their connection to the establishment of the Journal of Libertarian Studies is not clear.</think>
<answer>**Murray Newton Rothbard**</answer>
<analysis> One uncertainty arises from the fact that while [...], the information about him writing over 20 books on political theory is less direct. It would be better if [...].</analysis>
<confidence>**0.85**</confidence>

We use two variants of prompts. For Big-Math Digits, we use the *Simple Prompt*; for HotPotQA and HotPotQA-Vanilla, we use the *Long Prompt*, which provides additional guidelines for uncertainty analysis.

System Prompt (Simple)

A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The Assistant first thinks about the reasoning process in the mind, provides the user with the final answer, then analyzes its confidence about the solution and provides the user with its confidence level. The confidence level is a number between 0 and 1 (inclusive) enclosed within <confidence></confidence> tags. The final answer is enclosed between <answer></answer> tags. The analysis about confidence and uncertainty is enclosed within <analysis></analysis> tags. The Assistant should reason about its confidence in the solution and its uncertainty in the solution within these tags. The final format that must be followed is: <think> reasoning process here </think><answer> final answer here </answer><analysis> analysis about confidence and uncertainty here </analysis><confidence> confidence level here (number between 0 and 1) </confidence>

System Prompt (Long)

A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind, provides the user with the final answer, then analyzes its confidence about the solution and then provides the user with its confidence level. The confidence level is a number between 0 and 1 (inclusive) enclosed within <confidence></confidence> tags. The final answer is enclosed between <answer></answer> tags. The analysis about confidence and uncertainty is enclosed within <analysis></analysis> tags. The assistant should reason about its confidence in the solution and its uncertainty in the solution within these tags. Here are some guidelines for the analysis:

1. Your task is to point out things where the model could be wrong in its thinking, or things where there might be ambiguity in the solution steps, or in the reasoning process itself.
2. You should not suggest ways of fixing the response, your job is only to reason about uncertainties.
3. For some questions, the response might be correct. In these cases, it is also okay to have only a small number of uncertainties and then explicitly say that I am unable to spot more uncertainties.
4. Uncertainties might be different from errors. For example, uncertainties may arise from ambiguities in the question, or from the application of a particular lemma/proof.
5. If there are alternate potential approaches that may lead to different answers, you should mention them.
6. List out plausible uncertainties, do not make generic statements, be as specific about uncertainties as possible.
7. Enclose this uncertainty analysis within <analysis></analysis> tags.

The final format that must be followed is: <think> reasoning process here </think><answer> final answer here </answer><analysis> analysis about confidence and uncertainty here </analysis><confidence> confidence level here (number between 0 and 1) </confidence>

C.2 Verifiers

Verifiers are used during training to compute rewards and during evaluation to assess correctness. We employ three types of verifiers, following [11]:

1. **Exact-Match.** The predicted answer must exactly match the ground truth answer string. This verifier is used for HotPotQA and HotPotQA-Vanilla.

2. **Math-Verify.** We use `math-verify`, a robust mathematical expression evaluation system that checks semantic equivalence of mathematical expressions. This verifier is used for Math-500, GSM8K, and Big-Math Digits.
3. **LLM-as-a-Judge.** We use Llama-3.1-8B-Instruct with greedy decoding (temperature=0) as the judge model. The judge is provided with the question, the groundtruth answer, and the predicted answer, and is prompted to respond with "YES" or "NO" based on correctness. Importantly, we do not condition the judge on thinking traces to avoid potential biases. This verifier is used for TriviaQA, SimpleQA, CommonsenseQA, GPQA, and MedQA. The judge is served locally on fixed hardware with fixed software dependencies, so its output is (almost) deterministic. To rule out systematic errors introduced by a single judge, we also re-judge the out-of-domain generations with two additional judges from different model families and of different sizes (Sec. D.3).
4. **Code-Execution.** The generated program is executed against the test cases provided with each problem, and it is judged correct if all test cases pass. This verifier is used for MBPP.

LLM-as-a-Judge Prompt

You are given a question, a ground truth answer, and a predicted answer. Determine whether the predicted answer is correct based on the ground truth. Respond with only "YES" if the predicted answer is correct, or "NO" if it is incorrect.
Question: {question}
Ground Truth Answer: {ground_truth}
Predicted Answer: {predicted_answer}
Is the predicted answer correct?

C.3 LLM Response Parser

All responses use the `<think>`, `<answer>`, `<analysis>`, and `<confidence>` tags. The answers are extracted from the `<answer>` tags and confidence scores from the `<confidence>` tags. For models that do not follow the expected format, we append "Thinking time ended. My final answer is" to the output and prompt the model again to extract the answer. Similarly, if a valid confidence score cannot be extracted, we append "Thinking time ended. My verbalized confidence in my answer as a number between 0 and 1 is equal to" and prompt the model again. *Note that models trained with format rewards (see Sec. C.4.2) rarely trigger these fallbacks.*

C.4 Training

C.4.1 Training Datasets

1. **HotPotQA-Modified.** We adopt the HotpotQA-Modified dataset introduced by [11], which is derived from the original HotPotQA distractor dataset [26]. This modified version systematically varies the availability of supporting evidence by removing zero, one, or both of the key paragraphs required to answer each question. The questions are evenly distributed across these three conditions, and each instance contains 8 paragraphs in total. This construction creates varying levels of information completeness, making it particularly suitable for developing and evaluating uncertainty reasoning capabilities. The training set comprises $20,000$ examples, and correctness is evaluated by exact string matching. HotPotQA is distributed under CC BY-SA 4.0 License and HotPotQA-Modified is under MIT License. The convention of this paper uses HotPotQA for the modified version and HotPotQA-Vanilla for the original dataset.
2. **BigMath.** We use BigMath [27], a large-scale curated dataset for reinforcement learning that contains over $250,000$ mathematical problems. Following [11], we retain only problems with LLaMA-8B solve rates between 0% and 70% to maintain appropriate difficulty, and further restrict to problems with numerical answers to enable reliable automatic verification—resulting in the subset referred to as BigMath Digits. The final training set contains $15,000$ problems, with correctness evaluated using `math-verify`. It is distributed under the MIT License.

C.4.2 Training Details

Base LLM models & GRPO setup. We use Qwen2.5-7B [24] under the Apache 2.0 License and Llama3.1-8B [25] under the Llama 3.1 Community License. Following recent work on RL for reasoning [12, 48], we initialize RL directly from the base model without supervised finetuning

and set $\beta = 0$ (no KL regularization). We use GRPO without standard deviation normalization in the advantage calculation [34] *(an ablation confirms that the difference is minor: HotPotQA ID AURC 0.46 with std normalization vs. 0.44 without)*, along with the BNPO loss function [49], which aggregates token-level losses using the number of active tokens in the local training batch. For parameter-efficient training, we apply LoRA [50] with rank $r = 1$, $\alpha = 32$, and no dropout to all linear layers, motivated by *LoRA Without Regret* [51].

Reward functions. The total reward combines a task-specific reward with a format reward:

$$R_{\text{total}} = R_{\{\text{RLVR,RLCR,RLSR}\}} + R_{\text{format}}. \tag{C.1}$$

The method-specific reward is defined in Eq. (2.6) for RLVR, Eq. (2.7) for RLCR, and Eq. (2.15) for RLSR. The format reward $R_{\text{format}} \in \{0, 1\}$ is a binary indicator: 1 if the output follows the expected tag structure (`<think>`, `<answer>`, `<analysis>`, `<confidence>`) and contains a valid confidence score $c \in [0, 1]$, and 0 otherwise. *Note that the format reward converges to near-perfect values early in the training processes for all methods.*

Key hyperparameters. We generate 32 responses per prompt with temperature 0.7. **For HotPotQA**, we use a learning rate of 1×10^{-5} with a constant schedule and 5% warmup, training for 1 epoch with an effective batch size of 1536 (4 GPUs $\times$ 3 per-device batch $\times$ 128 gradient accumulation steps). The maximum prompt and completion lengths are 3072 and 1536 tokens, respectively. **For BigMath**, we use a learning rate of 5×10^{-5} with a linear decay schedule and 20% warmup, training for 0.5 epochs with an effective batch size of 1536 (4 GPUs $\times$ 6 per-device batch $\times$ 64 gradient accumulation steps). The maximum prompt and completion lengths are 1024 and 4096 tokens, respectively. All methods (RLVR, RLCR, RLSR) use identical hyperparameters to ensure fair comparison. All training uses the Transformers Reinforcement Learning (TRL) library, with vLLM as an efficient rollout engine [52]. Important system optimizations include `bfloat16` (`bf16`) precision, FlashAttention-2 (FA2) [53] and Triton kernels shipped with TRL.

Training time. Training runs on 4xNVIDIA H100 GPUs, and each run takes approximately 62 hours (HotPotQA) or 57 hours (BigMath). This holds for all three alignment methods, since they differ only in the reward computation, and the ranking step of RLSR takes negligible CPU time.

C.5 Evaluation

We follow RLCR [11] for setting up the evaluation protocol. All models are evaluated with greedy decoding (temperature=0) and a maximum token budget of 4096 (sufficient for all selected evaluation benchmarks). We generate a single response per question and use the same system prompts as during training: the simple prompt (Sec. C.1) for Big-Math Digits and the long prompt for HotPotQA.

C.5.1 Evaluation Datasets

1. **HotPotQA.** (MIT License) We use 500 held-out validation samples from the modified dataset introduced by [11]. Correctness is measured using exact-match.
2. **HotPotQA-Vanilla.** (CC BY-SA 4.0 License) We use $1,000$ validation samples from the original HotPotQA distractor dataset [26], with 2 irrelevant paragraphs removed from each question. Thus, each question contains 8 paragraphs with both supporting paragraphs present. Correctness is measured using exact-match.
3. **Math-500.** (MIT License) We use the MATH-500 dataset, a subset of problems from the original MATH dataset [31]. Correctness is measured using `math-verify`.
4. **GSM8K.** (MIT License) We use $1,319$ problems from the test set of the Grade School Math 8K dataset [32]. Correctness is measured using `math-verify`.
5. **Big-Math Digits.** (MIT License) We use $1,000$ held-out validation samples from the filtered Big-Math dataset [27]. Correctness is measured using `math-verify`.
6. **TriviaQA.** (Apache 2.0 License) We use $2,000$ samples from the TriviaQA [29] validation set, specifically the no-context split, to test the factual accuracy. Correctness is measured using LLM-as-a-judge.
7. **SimpleQA.** (MIT License) We use the full dataset consisting of $4,326$ factual questions [28]. Correctness is measured using LLM-as-a-judge.
8. **CommonsenseQA.** (MIT License) uses $1,220$ problems from the CommonsenseQA [33] validation set, a multiple-choice question answer dataset requiring various types of commonsense knowledge. Correctness is measured using LLM-as-a-judge.

9. **GPQA.** (MIT License) We use the main dataset (gpqa_main) that contains 448 multiple-choice questions written by experts in the fields of biology, physics, and chemistry [30]. Correctness is measured using LLM-as-a-judge.
10. **MedQA.** (MIT License) We use the 4-option version of MedQA [36], consistent with its original assessment and usage by technical reports of frontier medical LLMs [54, 55]. Correctness is measured using LLM-as-a-judge.
11. **MBPP.** (CC BY 4.0 License) We use 500 problems from the test split of the full version of MBPP [35], a benchmark of entry-level Python programming problems. Correctness is measured using code execution.

C.5.2 Evaluation Metrics

1. **AURC (↓).** The *area under risk-coverage curve* measures cumulative selective risk as a function of coverage with sorted prediction confidence. We refer the reader to Sec. 2.1 for the formal definition. Lower values means better SP performance.
2. **ECE (↓).** The *expected calibration error* measures the alignment between confidence and correctness likelihood:
$$\text{ECE} = \sum_{m=1}^{M} |B_m|/N \cdot |\text{acc}(B_m) - \text{conf}(B_m)| \tag{C.2}$$
where M is the number of bins, B_m is the set of samples in bin m, and N is the total number of samples. We use $M = 10$ bins.
3. **Accuracy@$k\%$ (↑).** Accuracy computed on the top $k\%$ most confident predictions, measuring how well the model's confidence identifies its correct predictions. We report results at coverage levels $k \in \{10, 25, 50\}$.

C.6 Mimicking Selective Prediction in Deployment

C.6.1 The MedQA Dataset

We use the 4-option version of MedQA [36], consistent with its original assessment and usage by frontier medical LLMs [54, 55]. Each question presents a clinical vignette followed by four answer choices (A, B, C, D). We partition the dataset as detailed in the following table:

Split	Samples	Usage
Train	10,178	alignment training
Validation	1,272	threshold selection
Test	1,273	deployment evaluation

The prediction correctness is checked by exact matching by the letter of the answer, i.e., among $\{A, B, C, D\}$. When the prediction does not follow the expected format, we apply robust answer extraction using multiple regex patterns: direct letter match, letter prefix (e.g., "D. Nitrofurantoin"), "The answer is" prefix, and pure-option-text-without-option-letter matching.

C.6.2 Inference Settings

We use the same generation settings as those of the in-domain evaluation (Sec. C.5): temperature set to 0, maximum token budget of 4096, and with the long system prompt (Sec. C.1). We use vLLM [52] for efficient inference. The models are loaded from HotPotQA training checkpoints.

C.6.3 Threshold Selection—A Key Step in Error-Controlled SP Deployment

Given a target accuracy α^* and the validation set D_{val}, we find the threshold τ via

$$\tau = \min\left\{t \in [0, 1] : \text{acc}(\{(x, y) \in D_{\text{val}} : s(x) \geq t\}) \geq \alpha^*\right\}, \tag{C.3}$$

where $s(x)$ is the confidence score for the input x. The corresponding coverage is computed as:

$$\text{coverage}(\tau) = |\{x \in D_{\text{val}} : s(x) \geq \tau\}|/|D_{\text{val}}|. \tag{C.4}$$

We set $\alpha^* = 75\%$.

C.6.4 Evaluation Metrics

After threshold selection above, we apply the threshold directly to the test set and report the following metrics:

- **Validation Accuracy**: the prediction accuracy on validation samples with confidence $\geq \tau$
- **Validation Coverage**: the fraction of validation samples with confidence $\geq \tau$
- **Test Accuracy**: the prediction accuracy on test samples with confidence $\geq \tau$
- **Test Coverage**: the fraction of test samples with confidence $\geq \tau$

C.6.5 Post-hoc Calibration Baseline

To evaluate whether post-hoc calibration can rescue poorly-calibrated confidence for SP, we apply temperature scaling [5] to RLVR's confidence output. Since our setting does not generate native logits, we follow [37] which proposes to calibrate a verbalized probability vector $(p_1, \ldots, p_K)$ via an invert-softmax trick: assume that the pre-softmax logits take the form $z \doteq (\log p_1 + b, \ldots, \log p_K + b)$ for an unknown constant b, the idea is to perform the temperature scaling $\mathrm{softmax}(z/T)$. For our single output case, we treat it as a binary classification problem on the $\{\text{correct}, \text{incorrect}\}$ classes, i.e., $K = 2$ so that the effective class probabilities are $(s, 1 - s)$. So:

$$s_T \doteq \mathrm{softmax}(z/T)_{\text{correct}} = \frac{e^{(\log s + b)/T}}{e^{(\log s + b)/T} + e^{(\log(1-s)+b)/T}} = \frac{1}{1 + e^{\frac{1}{T}\log(1-s)/s}} \tag{C.5}$$

$$= \mathrm{sigmoid}(\tfrac{1}{T} \log \tfrac{s}{1-s}), \tag{C.6}$$

yielding our temperature scaling operator. We find an appropriate T by training it over the binary calibration set $\{(s_i, y_i)\}_{i=1}^N$ (s_i: verbalized confidences; $y_i \in \{0, 1\}$: groundtruth correctness label):

$$T^* = \arg\min_{T>0} \; -\sum\nolimits_{i=1}^{N} y_i \log s_T(s_i) + (1 - y_i) \log(1 - s_T(s_i)). \tag{C.7}$$

The optimal $T^* \approx 47.9$ on the MedQA indicates severe overconfidence of the original prediction: it squashes the full probability domain $[0, 1]$ to $\approx [0.38, 0.62]$, and the confidence values the model actually emits, $\{0.5, \ldots, 1.0\}$, collapse into the even narrower band $\approx [0.50, 0.62]$.

It is easy to verify that S_T is a monotonically increasing function of s. Therefore, *it maintains the confidence ranking of the predictions—leaving the ranking-based risk-coverage curve unchanged*, as empirically confirmed in Fig. 7. The same holds for any monotonic scalar recalibration (e.g. Platt scaling). So, while confidence calibration does not necessarily improve SP performance, improving the confidence ranking is the key.

C.7 Experiment Details of Margin Analysis for the Lifted AURC Versus Canonical AURC

To compare the effect of lifted AURC versus AURC as the reward on the *margin*—the separation of correct and incorrect predictions in terms of confidence, we finetune the base model Qwen2.5-7B using an identical setting except for the reward function: one with lifted AURC (awarding correct predictions and penalizing incorrect predictions) and one with AURC (only penalizing incorrect predictions) on $6,000$ training samples of the HotPotQA dataset—enough for the format reward to converge for faithful parsing of confidence scores. We evaluate both models on the same $6,000$ training samples and 500 held-out test samples. For each sample, we extract the model's predicted answer, its correctness (determined by exact match), and its verbalized confidence score. We then partition the samples into correct and incorrect predictions and estimate their respective *normalized* distributions using kernel density estimation with a bandwidth of 0.15. Finally, we report the confidence gap, i.e., the difference between the mean confidence values of the two distributions. Fig. 3 visualizes these distributions, with solid lines denoting correct predictions and dotted lines denoting incorrect predictions.

D Additional Experiment Results

D.1 RLSR Works with Different Model Families

We follow the same training recipe and evaluation protocol as Tab. 1 for the Llama-3.1-8B model. The conclusion remains the same: RLSR achieves the best AURC (lower the better) performance averaged over in-domain (HotPotQA, HotPotQA-Vanilla) and out-of-domain (SimpleQA, TriviaQA, GPQA, Math500, GSM8K, CommonsenseQA) benchmarks.

Table 4: The Llama-3.1-8B results trained on HotPotQA. Orange highlights the best result.

Method	In-Domain AURC↓	Out-of-Domain AURC↓
BASE	0.54	0.41
RLVR	0.46	0.42
RLCR	0.43	0.40
RLSR	0.41	0.38

D.2 Statistical Significance of RLSR over RLCR and RLVR

Table 5: The AURC (mean $\pm$ std) of Qwen2.5-7B trained on HotPotQA over 5 seeds at $T = 0.3$.

Method	In-Domain AURC↓	Out-of-Domain AURC↓
RLVR	0.537 ± 0.008	0.453 ± 0.006
RLCR	0.509 ± 0.012	0.443 ± 0.009
RLSR	0.478 ± 0.006	0.412 ± 0.009

Our main results in Tab. 1 are reported using greedy decoding (temperature=0), where it is (almost) deterministic. To obtain variances and verify that the improvement of RLSR over the baselines is not due to sampling noise, we re-run all methods with 5 seeds at temperature 0.3 on HotPotQA. The non-overlapping confidence intervals in Tab. 5 confirm the statistical significance of RLSR's improved performance.

D.3 Robustness to the Choice of LLM Judge

Errors of the LLM judge directly affect AURC and accuracy-at-coverage. To rule out that the advantage of RLSR stems from systematic errors of a single judge, we re-judge the generations on the four OOD benchmarks that use LLM-as-a-Judge with two additional judges from different model families and of different sizes, Qwen2.5-32B-Instruct and gpt-oss-20b [56], both with greedy decoding. The absolute AURC varies between judges, but RLSR achieves the lowest AURC among all three judges (Tab. 6).

Table 6: AURC of Qwen2.5-7B trained on HotPotQA, averaged over the four OOD benchmarks that use LLM-as-a-Judge (TriviaQA, SimpleQA, CommonsenseQA, GPQA), under three judges. Orange highlights the best result per judge.

Judge	RLVR	RLCR	**RLSR**
Llama-3.1-8B-Instruct	0.51	0.50	0.46
Qwen2.5-32B-Instruct	0.59	0.58	0.56
gpt-oss-20b	0.61	0.60	0.59

D.4 Evaluation on Code Generation

Table 7: SP performance on MBPP, which is out-of-domain for both training settings, with the Qwen2.5-7B model. Orange highlights the best result per metric within each training setting.

	HotPotQA Training			**BigMath Training**		
Method	AURC↓	Acc@10↑	Acc@25↑	AURC↓	Acc@10↑	Acc@25↑
RLVR	0.37	62.8%	62.8%	0.43	57.2%	57.2%
RLCR	0.40	60.2%	61.0%	0.36	70.3%	64.8%
RLSR	0.34	67.3%	67.3%	0.32	77.4%	72.1%

To test generalization beyond QA and mathematical reasoning, we evaluate the final checkpoints of Tab. 1 on MBPP [35], a code generation benchmark used in the evaluation of Qwen2.5. Neither training set contains code, so MBPP is out-of-domain for both training settings. As shown in Tab. 7, RLSR attains the best AURC, Acc@10, and Acc@25 within each training setting, extending the out-of-domain advantage in Tab. 1 to a task type absent from both training sets.

D.5 Effect of Batch Size and Group Size

Table 8: Effect of the batch size B (unique prompts) and the group size G (rollouts per prompt) on Qwen2.5-7B trained on HotPotQA. Rollout cost counts the rollouts generated for the same training prompts, relative to the default setting. OOD Math: GSM8K and MATH-500; OOD Free-text: CommonsenseQA, GPQA, SimpleQA, and TriviaQA. All values are AURC. Orange highlights the best result per column.

						Out-of-Domain		
Method	B	G	Pool $B \times G$	Rollout Cost	In-Domain	Math (2)	Non-Math (4)	All (6)
RLVR	48	32	1536	1.0×	0.56	0.38	0.51	0.47
RLCR	48	32	1536	1.0×	0.51	0.37	0.50	0.46
RLSR	48	32	1536	1.0×	0.44	0.30	0.46	0.41
RLSR	32	32	1024	1.0×	0.44	0.49	0.46	0.47
RLSR	16	32	512	1.0×	0.45	0.50	0.47	0.48
RLSR	48	16	768	0.5×	0.46	0.35	0.45	0.42
RLSR	32	16	512	0.5×	0.47	0.39	0.44	0.43

RLSR ranks rollouts within a pooled batch of size $B \times G$ to approximate the population-level ranking in AURC, so a smaller pool introduces more finite-sample approximation noise. During training, the rollout cost depends only on G, while B determines the number of optimization steps for a fixed amount of training data. Tab. 8 shows that halving G, and thus the rollout cost, degrades AURC only slightly, and RLSR with $G = 16$ still outperforms RLVR and RLCR at the full cost in both ID and OOD evaluation. Reducing B to $G = 32$ leaves the ID and free-text OOD AURC nearly unchanged, but substantially degrades the AURC on the two math OOD benchmarks. We hypothesize that this is due to domain forgetting, a common failure mode in LLM finetuning: a smaller B leads to more optimization steps. The degradation is confined to the math benchmarks while ID performance is preserved, and following RLCR we train without KL regularization ($\beta = 0$). We have not tested whether RLVR and RLCR show similar behavior at smaller batch sizes.

D.6 Comparison with Supervised Finetuning (SFT) Baseline

To isolate the effect of RL training, we fine-tune Qwen2.5-7B on (question, answer) pairs from HotPotQA without chain-of-thought or confidence output because there is no groundtruth chain-of-thought and confidence. Since the SFT model cannot output verbalized confidence (VC), we use self-consistency (SC) with temperature=0.7 and samples $K = 8$ as a post-hoc confidence selector, where the fraction of completion agreement serves as the confidence score. We observe that SFT + SC (0.51) matches RLCR and outperforms RLVR, confirming that SC is a strong post-hoc selector. However, RLSR still outperforms it by 0.07 in AURC with a single inference pass, demonstrating the advantage of end-to-end RL training over supervised alternatives.

Table 9: Non-RL baseline comparison on HotPotQA with Qwen2.5-7B. SFT uses self-consistency (SC) as selector; RL methods use verbalized confidence (VC). Orange highlights the best result.

Method	Selector	ID AURC↓
Base	VC	0.60
SFT-Answer-Only	SC (K=8)	0.51
RLVR	VC	0.56
RLCR	VC	0.51
RLSR	VC	0.44

D.7 RLSR is Agnostic to the Choice of Confidence Selector

As the community has not converged to a standard for LLM confidence quantification [18–20], we use verbalized confidence (VC) as the default scorer throughout the paper for its simplicity and popularity. The RLSR reward in Eq. (2.15) depends on the confidence score s_i only through its rank r_i, so any confidence score that ranks rollouts can replace VC. To test whether the advantage of RLSR is unique to VC, we retrain RLSR and RLCR on HotPotQA with Qwen2.5-7B under two additional scorers, keeping the training hyperparameters of Sec. C.4.2 and removing the request for a VC from the system prompt: (1) length-normalized log-probability (Log-prob) of each rollout; and (2) self-consistency (SC), where the score of a rollout is the fraction of the $G = 32$ rollouts for the same prompt that share its answer. Rollouts with tied scores receive the average of their weights

Table 10: Matched-selector comparison of SP performance on two in-domain and six out-of-domain evaluation datasets with the Qwen2.5-7B model trained on HotPotQA. Each block fixes the selector and varies the alignment method. RLCR and RLSR use the selector in both training and evaluation; RLVR uses it at evaluation only. Orange highlights the best result per metric within each selector.

	In-Domain			Out-of-Domain		
Method	AURC↓	Acc@10↑	Acc@25↑	AURC↓	Acc@10↑	Acc@25↑
Verbalized Confidence (VC)						
RLVR	0.56	45.5%	42.0%	0.47	52.6%	52.9%
RLCR	0.51	53.5%	49.0%	0.46	53.5%	54.3%
RLSR	0.44	66.0%	63.2%	0.41	62.0%	60.7%
Length-Normalized Log-Probability (Log-prob)						
RLVR	0.55	56.0%	48.0%	0.48	49.7%	51.4%
RLCR	0.47	62.5%	58.2%	0.49	58.6%	56.2%
RLSR	0.47	63.5%	60.2%	0.46	55.9%	56.0%
Self-Consistency (SC, K=8)						
RLVR	0.37	78.5%	73.4%	0.38	71.8%	68.4%
RLCR	0.42	69.5%	68.4%	0.45	67.8%	60.9%
RLSR	0.35	82.0%	73.4%	0.37	73.6%	69.2%

$\widehat{\alpha}_i$. For RLCR, the Brier term uses the exponentiated Log-prob or the SC agreement fraction as s. Since the RLVR reward does not involve confidence, we do not retrain RLVR and instead evaluate its checkpoint from Tab. 1 with each selector. In evaluation, the SC confidence of a query is the fraction of $K = 8$ responses (one greedy, seven sampled in $T = 0.7$) that agree with the greedy answer, which serves as a prediction.

Tab. 10 shows that RLSR achieves the best or tied-best AURC under all three scorers in both ID and OOD evaluation. The margin is the largest with VC. With Log-prob, RLSR ties RLCR in ID AURC and trails it on OOD Acc@10 and Acc@25. With SC, RLVR is close to RLSR and ties it on ID Acc@25. RLSR with SC achieves the lowest overall AURC, at the cost of $K = 8$ forward passes per query at evaluation. We keep VC as default for three reasons. First, VC requires a single forward pass, whereas SC multiplies the inference cost by K. Second, VC is part of the sampled response, so the policy gradient trains the answer and its confidence jointly, without the additional architectural components required by the learned scorers [21, 22]. Third, VC follows the protocol of RLCR [11], which makes the comparison with RLCR direct, and produces a confidence value that can be reported to users.

D.8 Per-Benchmark Performance

Tab. 1 reports the average performance on the 8 benchmark datasets. Here, we include the performance per-benchmark for Qwen2.5-7B trained on HotPotQA (in-domain in Tab. 11 and out-of-domain in Tab. 12) and BigMath (in-domain in Tab. 13 and out-of-domain in Tab. 14) for completeness.

Table 11: The in-domain results of Qwen2.5-7B trained on HotPotQA.

Dataset	Method	AURC↓	ECE↓	Acc@10↑	Acc@25↑	Acc@50↑
HotPotQA	BASE	0.64	0.60	44.0	40.0	34.8
	RLVR	0.63	0.62	38.0	36.0	34.4
	RLCR	0.55	0.03	48.0	46.4	47.2
	RLSR	0.44	0.20	76.0	68.0	55.2
HotPotQA-Vanilla	BASE	0.57	0.54	48.0	41.2	40.8
	RLVR	0.50	0.51	53.0	48.0	49.2
	RLCR	0.46	0.07	59.0	51.6	53.6
	RLSR	0.44	0.27	56.0	58.4	59.2

Table 12: The out-of-domain results of Qwen2.5-7B trained on HotPotQA.

Dataset	Method	AURC↓	ECE↓	Acc@10↑	Acc@25↑	Acc@50↑
GSM8K	BASE	0.24	0.22	78.8	78.8	74.2
	RLVR	0.21	0.20	78.8	80.0	79.2
	RLCR	0.17	0.33	80.3	83.6	84.1
	RLSR	0.16	0.11	81.8	85.8	87.9
MATH-500	BASE	0.51	0.50	50.0	51.2	46.4
	RLVR	0.56	0.55	42.0	43.2	42.4
	RLCR	0.56	0.06	42.0	45.6	46.8
	RLSR	0.45	0.32	66.0	57.6	53.2
CommonsenseQA	BASE	0.08	0.01	92.7	92.8	92.5
	RLVR	0.13	0.11	86.2	86.3	88.4
	RLCR	0.08	0.40	95.1	92.5	91.7
	RLSR	0.09	0.14	94.3	92.8	90.8
GPQA	BASE	0.55	0.48	53.3	49.1	44.2
	RLVR	0.60	0.60	44.4	39.3	39.7
	RLCR	0.66	0.15	24.4	32.1	37.9
	RLSR	0.58	0.35	46.7	45.5	42.4
SimpleQA	BASE	0.86	0.81	14.5	13.8	13.6
	RLVR	0.89	0.89	9.7	11.7	11.1
	RLCR	0.88	0.36	11.8	11.4	11.5
	RLSR	0.85	0.50	12.9	14.1	14.8
TriviaQA	BASE	0.40	0.37	65.0	58.6	58.6
	RLVR	0.42	0.42	54.5	56.8	57.8
	RLCR	0.38	0.09	67.5	60.8	60.0
	RLSR	0.34	0.27	70.5	68.4	67.8

Table 13: The in-domain results of Qwen2.5-7B trained on BigMath.

Dataset	Method	AURC↓	ECE↓	Acc@10↑	Acc@25↑	Acc@50↑
BigMath	BASE	0.49	0.46	48.0	54.4	53.0
	RLVR	0.37	0.32	56.0	63.2	64.6
	RLCR	0.32	0.07	69.0	69.6	68.0
	RLSR	0.24	0.19	79.0	80.8	75.2
GSM8K	BASE	0.24	0.22	78.8	78.8	74.2
	RLVR	0.12	0.10	88.6	87.3	88.2
	RLCR	0.13	0.27	87.1	86.4	86.5
	RLSR	0.12	0.07	89.4	90.3	88.0
MATH-500	BASE	0.51	0.50	50.0	51.2	46.4
	RLVR	0.36	0.34	72.0	61.6	61.6
	RLCR	0.33	0.03	74.0	70.4	66.4
	RLSR	0.36	0.29	68.0	64.0	66.0

Table 14: The out-of-domain results of Qwen2.5-7B trained on BigMath.

Dataset	Method	AURC↓	ECE↓	Acc@10↑	Acc@25↑	Acc@50↑
HotPotQA-Vanilla	BASE	0.57	0.54	48.0	41.2	40.8
	RLVR	0.53	0.53	51.0	44.8	46.6
	RLCR	0.47	0.12	63.0	55.2	53.2
	RLSR	0.49	0.36	55.0	55.6	51.8
CommonsenseQA	BASE	0.08	0.01	92.7	92.8	92.5
	RLVR	0.15	0.07	83.7	83.7	84.6
	RLCR	0.11	0.29	90.2	89.9	88.9
	RLSR	0.09	0.10	91.1	91.8	92.6
GPQA	BASE	0.55	0.48	53.3	49.1	44.2
	RLVR	0.59	0.59	42.2	37.5	40.2
	RLCR	0.57	0.18	48.9	45.5	39.7
	RLSR	0.53	0.35	53.3	48.2	45.1
SimpleQA	BASE	0.86	0.81	14.5	13.8	13.6
	RLVR	0.89	0.86	10.9	10.6	10.4
	RLCR	0.88	0.44	12.9	12.2	13.1
	RLSR	0.87	0.65	14.5	12.5	12.4
TriviaQA	BASE	0.40	0.37	65.0	58.6	58.6
	RLVR	0.50	0.40	44.0	43.0	52.0
	RLCR	0.33	0.04	71.5	66.4	65.6
	RLSR	0.30	0.22	78.5	73.6	69.1

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We elucidate our contributions in the abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitation in Sec. 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: Our study is supported by empirical results only.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We report detailed training and evaluation settings in Sec. 4.1 and Sec. C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The data is public. The code is at `https://github.com/sun-umn/RLSR`.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We report detailed training and evaluation settings in Sec. 4.1 and Sec. C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The main results are evaluated via greedy decoding from LLMs, but we include a temperature variant for statistical significance in Sec. D.2.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the compute resources in Sec. C.4.2.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read this code.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader impacts in Sec. 5.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification: The models are not exposed to new harmful information other than what the base models have already learned. The data is also free from high-risk content.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We discuss the license in Sec. C.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Related code is released under MIT license.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [No]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs are the subject of the method, and LLM-as-a-Judge is part of the evaluation. Regarding presentation, LLM is used only for polishing and making visuals.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.