SOPO: Socratic Guided Policy Optimization for Span-Level Hallucination Detection

Jiawei Dong[1] **Zilong Bai**[1] **Pengtian Zhu**[1] **Peng Zhou**[1]
[1]Patsnap Pte. Ltd., Suzhou, China
{dongjiawei,baizilong,zhupengtian,tonyzhoupeng}@patsnap.com

Abstract

Large Language Models (LLMs) have gained widespread adoption in various natural language processing tasks, but suffer from hallucination issues where they generate unfaithful or inconsistent content. While Reinforcement Learning with Verifiable Rewards (RLVR) has shown promising applications in reasoning tasks, its application to span-level hallucination detection is challenged by localization ambiguity, reward sparsity on hard examples, and uncontrolled length inflation. To address these issues, we introduce Socratic Guided Policy Optimization (SOPO), a multi-turn RL framework for robust hallucination detection. The key innovation of SOPO is a Socratic feedback mechanism, where a teacher model guides the reasoning trajectories during rollout by providing indirect hints inferred from the ground truth. This teacher-guided approach enables the model to generate more effective reasoning trajectories for hard examples, thus alleviating reward sparsity. Furthermore, we incorporate Turn-based Advantage Scaling to penalize verbosity bias, encouraging the model to develop efficient and autonomous reasoning capabilities. Empirical results on RAGTruth demonstrate that SOPO (4B) achieves state-of-the-art performance, surpassing leading proprietary models (e.g., GPT-5) and standard RL baselines, while obtaining higher reported mean scores than GRPO on four out-of-domain benchmarks, with mixed evidence against CAPO. We release our training and evaluation code alongside the 1.7B and 4B SOPO models at `https://github.com/patsnap/SOPO`.

1 Introduction

Retrieval-Augmented Generation (RAG) has emerged as a dominant paradigm for mitigating hallucinations in Large Language Models (LLMs) by grounding generation in external knowledge. However, even with retrieved context, LLMs inevitably produce factually inconsistent content [21, 31]. Detecting these hallucinations at a fine-grained level, specifically by pinpointing the exact hallucinated spans, is critical for ensuring the reliability and safety of real-world applications.

While Supervised Fine-Tuning (SFT) has established a baseline for this task, recent research has shifted towards Reinforcement Learning (RL), particularly outcome-based methods like Reinforcement Learning with Verifiable Rewards (RLVR), to further align models with complex reasoning requirements.

Prior research has largely focused on case-level (binary) hallucination detection using SFT or RL [4]. For instance, FaithLens [18] demonstrated that an 8B model, trained via a rigorous four-stage pipeline, could rival proprietary models on case-level tasks. To address the granularity gap, RL4HS [20] pioneered the application of RL to span-level detection, utilizing Class-Aware Policy Optimization (CAPO) to outperform proprietary models.

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

While recent guided-RL and iterative-refinement paradigms aim to enhance reasoning, they exhibit structural mismatches for objective, fine-grained hallucination detection. Specifically, unified models [25] compromise diagnostic objectivity due to role conflicts when evaluating their own generations; Existing Socratic-RL frameworks [24] rely on post-hoc distillation instead of providing dynamic, instance-specific hints during active RL exploration; and direct feedback methods [29] supply explicit single-turn corrections, causing ground-truth leakage that bypasses autonomous reasoning process.

Consequently, in the absence of suitable guided frameworks, standard RLVR continues to serve as the default baseline. Yet, our analysis demonstrates that applying such scalar-reward RL directly to fine-grained span detection is inherently hindered by three fundamental limitations:

Ambiguity of Localization Signals. From a reward modeling perspective, systematic analyses of RLHF have highlighted the fundamental limitations of scalar reward models regarding expressiveness and feedback sparsity [2]. In the specific context of span-level hallucination detection, this limitation is amplified: a single scalar reward is forced to represent a complex, comprehensive evaluation of accuracy across three distinct dimensions, namely boundary alignment, semantic completeness, and factual discrimination. Consequently, using such a coarse-grained proxy for a fine-grained task introduces significant ambiguity. For instance, particularly in samples where the target span consists of only a few words, a deviation of a single token can trigger drastic drops in the reward score (e.g., F1 dropping from 1.0 to 0.2), resulting in instability in the training process. This instability is supported in RL4HS [20], where the direct application of span-F1 as a reward required the introduction of CAPO to mitigate reward imbalance. This evidence further confirms that coarse scalar signals induce optimization instability and bias in hallucination span detection.

Reward Sparsity on Hard Examples. The model struggles to learn from hard examples due to the sparsity of the reward signal. With a small group size (G), the model consistently receives a zero reward across epochs for difficult queries, causing performance plateauing in the later stages of training. Conversely, simply increasing G to improve exploration introduces a trade-off: it generates a large number of "zero-advantage" groups for easy samples (where all rollouts are correct), which provides negligible gradient information [30, 12]. This results in wasted computational resources and significantly reduces overall training efficiency [7].

Uncontrollable Reasoning Length Inflation. During RLVR training, length of reasoning chains exhibits a tendency to inflate uncontrollably as training progresses. This phenomenon is not merely accidental but appears to be a structural limitation of current RL objectives. Prior analyses suggest that this verbosity is largely an optimization artifact: PPO/GRPO-style objectives systematically drive the policy toward longer outputs, particularly when the model struggles with correctness, leading to a persistent preference for lengthy generations [6, 3]. This aligns with recent findings on "systematic length inflation," where models trade off efficiency for length by generating redundant "filler" tokens without improving accuracy [17]. Consequently, without explicit regularization, this length inflation is bounded only by the context window. For practical applications, such verbosity creates severe computational bottlenecks and drastically increases inference latency.

To address these limitations, drawing inspiration from human pedagogy, we propose **Socratic Guided Policy Optimization (SOPO)**. Central to SOPO is a teacher LLM that engages the student in a multi-turn Socratic dialogue, offering step-wise analysis and guiding questions based on the student's incorrect reasoning steps *without revealing the ground truth*. This iterative interaction transforms sparse outcome rewards into dense, constructive supervision. Complementing this, we introduce a Turn-based Advantage Scaling mechanism that explicitly penalizes verbosity and reliance on multi-turn feedback, encouraging the model to achieve efficient, autonomous detection in the initial turn.

We validate the effectiveness of our approach on the RAGTruth benchmark [13]. Empirical evaluations demonstrate that SOPO establishes a new state-of-the-art with remarkable training efficiency.

Our main contributions are summarized as follows:

- We identify the limitations of vanilla RLVR in span-level hallucination detection task and propose SOPO, a framework that incorporates a Socratic-style feedback mechanism. By offering analysis and guiding questions without ground truth leakage, we transform the

training process from passive reward reception to active iterative reasoning refinement, effectively preventing shortcut learning while encouraging intrinsic self-correction.

- We introduce Turn-based Advantage Scaling, a simple yet effective technique that explicitly penalizes verbosity and reliance on multi-turn feedback. This mechanism reshapes the advantage to encourage autonomous and efficient reasoning, addressing the issue of reasoning length inflation common in RL training.
- Experiments on the RAGTruth and other out-of-distribution benchmark demonstrate that SOPO (4B) achieves superior performance. Notably, despite its compact size (4B parameters), SOPO outperforms larger proprietary models (e.g., GPT-5, o3) and surpasses the previous state-of-the-art by 4.4 points in overall F1 on RAGTruth test set, implying the effectiveness of our Socratic RL paradigm.

2 Span-level Hallucination Detection

2.1 Task

We focus on the task of Span-level Hallucination Detection within Retrieval-Augmented Generation (RAG) scenarios, encompassing tasks such as summarization and knowledge-intensive question answering. Formally, let $\mathcal{C}$ denote the retrieved context document and $\mathcal{Y} = (y_1, y_2, \ldots, y_T)$ denote the model-generated response consisting of T tokens or characters. The objective is to identify a set of unreliable spans $\hat{\mathcal{S}} = \{s_1, s_2, \ldots, s_n\}$, where each span s_k represents a sequence within $\mathcal{Y}$ that contradicts or is unsubstantiated by the context $\mathcal{C}$. Each span is defined by its start and end character indices, i.e., $s_k = [start_k, end_k)$. If the response contains no hallucinations, $\hat{\mathcal{S}}$ should be an empty set.

2.2 Metric

To rigorously evaluate the localization accuracy of the detected hallucinations, we adopt the Span-level F1 score, a standard metric for span-level hallucination detection task [13].

Let $\mathcal{S}^*$ represent the set of ground-truth hallucination character indices, and $\hat{\mathcal{S}}$ represent the set of predicted character indices. We treat the spans as sets of integers representing character positions. The Precision (P) and Recall (R) are calculated based on the character-level overlap between the prediction and the ground truth, where $|\cdot|$ denotes the cardinality of the set (i.e., the total number of characters), and $\cap$ denotes the intersection operation. The final metric is given by:

$$P = \frac{|\hat{\mathcal{S}} \cap \mathcal{S}^*|}{|\hat{\mathcal{S}}|}, \quad R = \frac{|\hat{\mathcal{S}} \cap \mathcal{S}^*|}{|\mathcal{S}^*|}, \quad \text{span-F1} = \frac{2 \cdot P \cdot R}{P + R} \tag{1}$$

This metric encourages the model to cover the hallucinated content comprehensively (high recall) while minimizing the inclusion of factual text (high precision).

3 SOPO: Socratic Guided Policy Optimization

To address the aforementioned limitations of RLVR on hallucination detection, we introduce a heuristic multi-turn training strategy named **Socratic Guided Policy Optimization (SOPO)**. By incorporating a Teacher LLM, SOPO shifts the paradigm from scalar reward maximization to iterative refinement. At each turn, the teacher evaluates the student's reasoning and generates a detailed analysis of specific incorrect steps, and then raises a guiding question to encourage the student to self-correct. Through this multi-turn, Socratic-style feedback, we guide the model toward dense, step-wise optimization, thereby alleviating the performance plateauing often observed under sparse reward settings. The overall workflow is shown in Figure 1.

3.1 Formalization of Socratic Interaction

We formulate the hallucination detection task as a multi-turn decision-making process involving a student policy π_θ and a frozen teacher policy $\pi_{teacher}$. Let x denote the input context (including

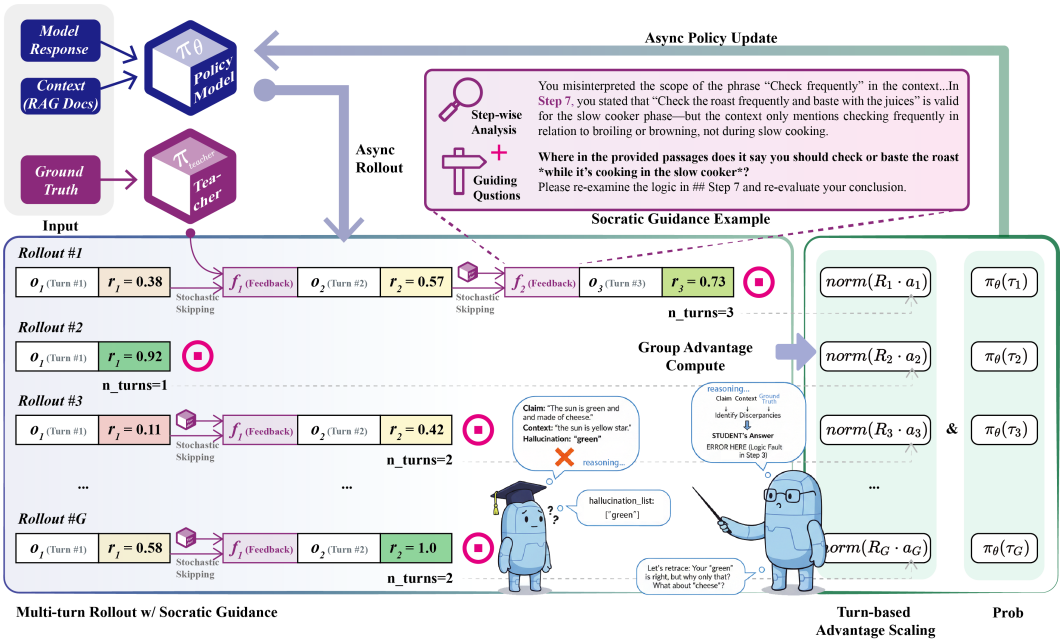

Figure 1: Overview of SOPO.

source documents and the response to be verified) and y^* represent the ground-truth hallucination spans.

The turns of interaction is limited to T. At each turn t ($1 \leq t \leq T$), the process consists of three phases: generation, evaluation, and feedback.

Student Generation. Given the input x and the interaction history H_{t-1}, the student model generates a reasoning path and a prediction o_t:

$$o_t \sim \pi_\theta(\cdot \mid x, H_{t-1}) \tag{2}$$

where $H_0 = \emptyset$, and H_{t-1} accumulates the trajectory of previous outputs and feedback.

Evaluation. At each turn, we evaluate the quality of o_t against the ground truth y^* using the span-level F1 reward function $R(\cdot)$. The interaction process terminates early if the reward exceeds a predefined threshold δ (e.g., $\delta = 0.6$), serving as a regularization measure to prevent overfitting to potential annotation noise in the training set. Furthermore, to prevent the student policy from developing a rigid dependency on teacher's feedback, we stochastically provide feedback: for lower-scoring responses ($R(o_t, y^*) < \delta$), the teacher provides feedback with probability p_{feedback}; otherwise, the feedback phase is skipped.

Socratic Feedback Generation. The teacher model $\pi_{teacher}$ acts as a Socratic guide, unlike standard correction methods that directly provide y^*, the teacher generates a feedback signal f_t conditioned on the student's incorrect reasoning, specifically designed to be *answer-agnostic*, encouraging the student to derive the conclusion independently. The feedback consists of two components: a step-wise analysis ($\mathcal{A}$) and a guiding question ($\mathcal{G}$):

$$f_t = [\mathcal{A}_t; \mathcal{G}_t] \sim \pi_{teacher}(\cdot \mid x, o_t, \mathcal{P}_{\text{Socratic}}) \tag{3}$$

where $\mathcal{P}_{\text{Socratic}}$ represents the system prompt enforcing the constraints against leaking the ground truth.

The history is then updated to $H_t = H_{t-1} \cup \{o_t, f_t\}$, and the student attempts to refine its reasoning in the next turn $t+1$. The complete trajectory τ for a sample is thus defined as:

$$\tau = \{(o_1, r_1), (f_1), (o_2, r_2), (f_2), \ldots, (f_{k-1}), (o_k, r_k)\} \tag{4}$$

where r_t is the reward score of o_t and $k \leq T$ is the turn where the model either succeeds or reaches the turn limit. Our optimization goal is to maximize the reward of these trajectories, weighted by our turn-based advantage scaling mechanism.

3.2 GRPO with Turn-based Advantage Scaling

We adopt Group Relative Policy Optimization [16] as our reinforcement learning backbone. Unlike PPO, GRPO eliminates the explicit value function critic and estimates the baseline directly from group scores. The learning objective is defined as:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[\min\left(r_\theta(\tau)A(\tau), \text{clip}\left(r_\theta(\tau), 1-\varepsilon, 1+\varepsilon\right)A(\tau)\right)\right] \tag{5}$$

where τ denotes a trajectory sampled from π_θ, and $r_\theta(\tau)$ is the probability ratio between the updated and reference policies.

In original GRPO, the advantage A_i for the i-th output in a group of size G is typically computed by directly normalizing the raw rewards $\mathbf{R} = \{R_1, \ldots, R_G\}$:

$$A_i = \frac{R_i - \text{mean}(\mathbf{R})}{\text{std}(\mathbf{R}) + \epsilon} \tag{6}$$

However, relying solely on unpenalized task rewards fails to distinguish and penalize trajectories that achieve identical scores but consume significantly more interactions. This flaw inadvertently encourages interaction redundancy, allowing the model to secure high rewards by heavily relying on extended dialogue rather than solving the task autonomously.

To counteract this, we introduce a Turn-based Advantage Scaling mechanism. Given a sampled trajectory τ_i consisting of a total of k_i turns ($1 \leq k_i \leq T$), we define a penalty hyperparameter p_{penalty} to compute a scaling factor α_i, which modifies the advantage as follows:

$$\begin{aligned} \alpha_i = \text{clip}\left(1 - (k_i - 1) \cdot p_{\text{penalty}}, 0, 1\right), \quad R_{\text{scaled},i} = \alpha_i \cdot R_i, \\ A_i = \frac{R_{\text{scaled},i} - \text{mean}(\mathbf{R}_{\text{scaled}})}{\text{std}(\mathbf{R}_{\text{scaled}}) + \epsilon} \end{aligned} \tag{7}$$

where $\mathbf{R}_{\text{scaled}} = \{R_{\text{scaled},1}, \ldots, R_{\text{scaled},G}\}$ represents the set of scaled rewards within the group. This pre-normalization intervention effectively incorporates our heuristic constraints without violating the mathematical integrity of the zero-mean advantage estimator (see Appendix H). With these modifications, we explicitly penalize reliance on multi-turn feedback and encourage autonomous reasoning without altering the fundamental definition of the task metric.

3.3 Span-Level F1 Reward

To align with our fine-grained objective, we formulate the reward directly based on the Span-level F1 score. Let $\hat{\mathcal{S}}$ denote the set of predicted hallucination spans and $\mathcal{S}^*$ represent the ground-truth spans. The reward at turn t is:

$$r_t = \begin{cases} 1, & \text{if } \hat{\mathcal{S}} = \emptyset \text{ and } \mathcal{S}^* = \emptyset, \\ \text{span-level F1}(\hat{\mathcal{S}}, \mathcal{S}^*), & \text{otherwise.} \end{cases} \tag{8}$$

In multi-turn training, we employ a turn-averaged reward to encourage early error correction:

$$R_i = \frac{1}{k_i}\sum_{t=1}^{k_i} r_t \tag{9}$$

where k_i is the total number of turns in trajectory τ_i. Complementary to Advantage Scaling, this acts as a local quality constraint to prevent the model from intentionally failing early turns to solicit hints (detailed in Appendix J).

4 Experimental Setup

Dataset. To directly validate the effectiveness of our proposed method in Retrieval-Augmented Generation (RAG) scenarios, we conduct experiments on the RAGTruth [13] benchmark. This dataset encompasses three distinct tasks: Summarization, Question Answering (QA), and Data-to-Text. The training set contains 15,090 samples, and the test set contains 2,700 samples. Each sample in the dataset consists of paired source documents, LLM-generated response, and human-annotated hallucination spans.

Model. In our experiments, we use the Qwen3-4B-2507 reasoning model [26] as the policy model and the Qwen3-Next-80B-A3B-Instruct as the feedback model. The parameters of the feedback model remain fixed throughout all experiments.

Cold-start. To accelerate convergence and address the cold-start problem, we randomly split the RAGTruth training set into a 5,090-sample subset for SFT and a 10,000-sample subset for RL training. We employ Gemini-2.5-Flash to generate multiple responses for each SFT prompt. Then, we apply rejection sampling to select responses with the highest Span-level F1 scores against the ground truth to construct the cold-start SFT dataset. Both SOPO and all baseline models are initialized from this SFT checkpoint for the subsequent RL training.

Table 1: **Span-level hallucination detection results on RAGTruth.** We compare our results with several approaches presented in Multi-View [14], RL4HS [20], LettuceDetect [32]. Best scores are in bold, second best scores are underlined.

Model	Summarization			Question Answering			Data-to-Text			Overall		
	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall
Proprietary models												
GPT-4o-mini w/ CoT	38.4	43.4	34.4	27.3	33.7	23.0	33.7	34.2	33.2	32.4	35.9	29.5
GPT-5-mini w/ CoT	43.9	33.0	65.5	47.2	37.9	62.7	42.5	29.8	**74.7**	44.2	32.8	67.7
GPT-5 w/ CoT	36.5	24.9	**68.4**	44.4	32.1	71.8	45.7	33.2	73.5	42.9	30.7	**71.1**
o3 w/ CoT	48.5	40.7	60.1	49.9	43.4	58.9	55.2	45.6	70.0	51.8	43.8	63.2
Non-Reasoning models												
Qwen3-4B-Instruct-2507	43.4	46.6	40.7	33.7	27.4	43.8	35.7	28.3	48.3	36.2	30.2	45.0
Qwen3-30B-A3B-Instruct-2507	35.1	37.5	33.0	19.0	20.4	17.8	28.5	29.1	28.0	26.5	27.7	25.3
Qwen3-32B-Instruct	44.0	59.7	34.9	19.4	33.3	13.7	34.4	44.3	28.2	31.5	44.7	24.3
Qwen3-Next-80B-A3B-Instruct	48.2	45.7	51.1	33.8	26.6	46.3	42.5	39.5	45.9	39.9	34.5	47.2
Reasoning models												
Qwen3-8B	34.7	42.2	29.5	30.5	32.0	29.1	20.3	45.2	13.1	27.8	37.2	22.1
Qwen3-14B	35.8	36.9	34.9	30.6	30.7	30.6	34.8	40.9	30.4	33.3	35.8	31.1
Qwen3-4B-2507	47.0	44.0	50.4	34.7	33.3	36.1	49.8	48.0	51.7	43.6	41.8	45.7
Qwen3-30B-A3B-2507	46.9	43.8	50.6	37.2	33.7	41.5	51.0	51.7	50.3	44.8	42.6	47.1
Span-level Hallucination Detection												
Multi-View Attention 7B	41.5	49.6	35.7	50.6	38.5	**73.7**	55.2	53.5	57.1	50.7	45.1	57.9
RL4HS-CAPO-7B	50.9	64.4	42.3	56.4	57.1	56.5	60.4	67.1	54.9	56.9	62.4	52.3
RL4HS-CAPO-14B	**57.6**	64.2	52.3	54.8	52.5	57.3	62.6	67.2	58.7	58.3	60.5	56.3
LettuceDetect-Base	36.7	53.0	28.1	61.5	62.7	60.4	57.4	58.2	56.6	55.4	59.4	52.0
LettuceDetect-Large	44.6	60.2	35.5	**64.4**	**66.9**	62.1	60.0	64.7	56.0	58.9	64.9	54.0
Baseline-SFT-4B	52.3	63.5	44.5	59.5	53.3	67.4	60.5	60.1	61.0	58.7	57.5	59.9
Baseline-GRPO-4B	53.2	**70.9**	42.6	59.3	62.8	56.1	64.0	**67.9**	60.4	60.2	66.4	55.1
Baseline-CAPO-4B	53.2	66.1	44.6	60.1	58.0	62.3	63.7	64.8	62.6	60.4	62.2	58.7
SOPO-4B	55.7	70.1	46.2	63.9	66.4	61.6	**66.0**	67.1	65.0	**63.3**	**67.3**	59.8

Training Details. Our RL framework is implemented using AREAL [8] along with dynamic sampling [28] to accelerate the training process. All experiments are conducted on 20 NVIDIA A800 GPUs. We employ a decoupled resource strategy, allocating 8 GPUs for rollouts, 8 for updates and 4 for feedback model inference. Full hyperparameters are listed in Table 7.

Inference. Consistent with the baselines, all methods (including SOPO) perform single-turn inference on the RAGTruth test set: given the retrieval-augmented context and the LLM-generated response, the model directly produces the reasoning chain and the predicted hallucination spans without any multi-turn interaction or teacher feedback.

Table 2: **Case-level hallucination detection results on RAGTruth.** We compare our results with several approaches presented in Ragas [5], Trulens, Luna [1], LettuceDetect [32], Grounded in Context [9] and RAG-HAT [19]. Best scores are in bold, second best scores are underlined.

Model	Summarization			Question Answering			Data-to-Text			Overall		
	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall
Ragas	40.8	64.2	29.9	35.7	31.2	41.9	61.9	79.2	50.8	52.0	62.0	44.8
Trulens	44.5	40.2	50.0	36.6	22.8	**92.5**	79.0	66.9	**96.5**	60.4	46.5	85.8
Luna	52.5	40.0	76.5	51.3	37.8	80.0	75.9	64.9	91.2	65.4	52.7	86.1
LettuceDetect-Base	50.5	53.9	47.6	65.5	60.6	71.3	87.9	89.3	86.5	76.1	76.6	75.5
LettuceDetect-Large	59.7	64.0	55.9	70.2	65.9	75.0	88.5	90.5	86.7	79.2	80.4	78.1
Grounded in Context	**86.2**	**85.4**	**87.1**	**90.1**	**88.4**	91.9	72.2	64.4	82.2	83.1	79.4	**87.1**
RAG-HAT	67.6	77.7	59.8	74.8	76.5	73.1	91.6	**92.9**	90.3	83.9	87.3	80.8
Baseline-SFT-4B	73.6	76.3	71.1	73.0	69.5	76.9	91.7	89.6	93.8	84.7	83.4	86.0
Baseline-GRPO-4B	69.5	83.0	59.8	71.7	74.8	68.8	91.7	92.5	91.0	84.0	87.9	80.5
Baseline-CAPO-4B	70.5	76.1	65.7	71.8	66.0	78.8	91.8	90.5	93.3	83.9	83.0	84.8
SOPO-4B	71.9	81.0	64.7	72.7	78.8	67.5	**92.0**	92.2	91.9	**84.8**	**88.0**	81.9

5 Results

5.1 Main Results

Comprehensive Superiority on Span-level Detection. As presented in Table 1, SOPO (4B) establishes a new state-of-the-art in overall F1 score on the RAGTruth test set, significantly outperforming larger proprietary models (e.g., GPT-5, o3). Notably, it surpasses the previous best framework, LettuceDetect-Large, by **4.4 points** in overall F1 (63.3 vs. 58.9). SOPO also outperforms all baseline models trained by other methods (SFT, GRPO, and CAPO) across all three tasks (note that CAPO/RL4HS's original scores are calibrated for fair comparison, detailed in Appendix I). Crucially, SOPO alleviates the aforementioned reward sparsity, maintaining robust performance even on highly constrained hard examples (Appendix G).

Generalization to Case-level Detection. Although optimized for span-level granularity, SOPO generalizes effectively to case-level detection. As shown in Table 2, it achieves a state-of-the-art Overall F1 of 84.8, surpassing dedicated methods (e.g., RAG-HAT) and all standard RL baselines. Notably, SOPO excels in Data-to-Text (92.0 F1) and yields the highest overall precision (88.0).

Inference-time significance. We additionally evaluated the full SOPO checkpoint and the same-start CAPO and GRPO baselines over 10 full RAGTruth test-set inference runs per method. The mean overall span-F1 is 63.34 for SOPO, 59.75 for CAPO, and 59.49 for GRPO. The reported paired bootstrap differences are +3.59 points versus CAPO (nominal 95% CI [1.52, 5.75], one-sided $p = 0.0005$) and +3.85 versus GRPO ([1.33, 6.57], $p = 0.0008$). For overall case-F1, the corresponding means are 84.65 versus 83.03 for both baselines, with nominal one-sided $p = 0.0209$ and $p = 0.0248$, respectively. The complete results and bootstrap protocol are reported in Appendix E.

Breakthrough on Hard Examples. Addressing the reward sparsity issue on difficult queries identified in Section 1, SOPO demonstrates a significant breakthrough on hard examples. As detailed in Appendix G, SOPO consistently improves prediction scores across samples with extreme constraints, including overly short hallucinated spans, dense multi-span occurrences, and top-10% longest contexts. By transforming sparse outcome rewards into dense Socratic guidance, SOPO successfully drives a robust positive shift in the score distribution compared to GRPO, directly contributing to its superior overall performance.

Out-of-Domain Evaluation. Across 10 inference runs on each of four out-of-domain (OOD) benchmarks, SOPO's mean primary score exceeds GRPO's: 74.07 vs. 72.71 span-F1 on PsiloQA, 89.13 vs. 86.19 span-F1 on Math-Shepherd-614, 75.98 vs. 73.64 mean balanced accuracy on LLM-AggreFact, and 69.78 vs. 67.89 accuracy on HaluEval. The nominal bootstrap interval versus GRPO crosses zero on PsiloQA; comparisons with CAPO are mixed. Detailed run-pool and sample-paired bootstrap uncertainty estimates for each benchmark are provided in Appendix F.

Accelerated Convergence and Autonomous Reasoning. As illustrated in Figure 2, SOPO demonstrates superior training efficiency, achieving a task reward at $\sim$200 steps that CAPO requires $\sim$600 steps to achieve. This rapid convergence is supported by evaluation results in Figure 4, where SOPO

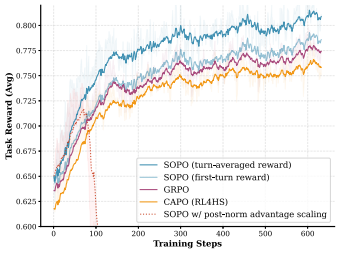

Figure 2: Mean task reward of RL over training steps.

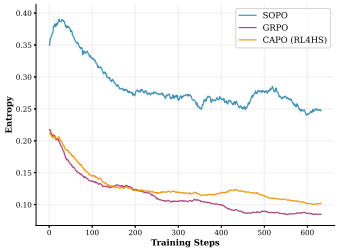

Figure 3: Entropy of RL over training steps.

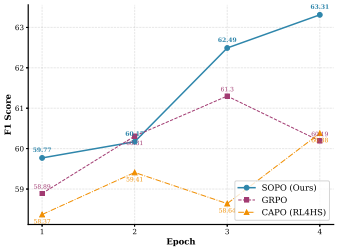

Figure 4: Overall F1 on RAGTruth test set over training epochs.

matches GRPO's peak performance (4 epochs) within just two epochs. Crucially, this efficiency stems from enhanced autonomous reasoning: as shown in Figure 2, SOPO's performance in the first turn is consistently higher than other baselines even in the first epoch. This confirms that our turn-based advantage scaling mechanism successfully encourages the model to reduce reliance on teacher's feedback, driving it to perform accurate hallucination detection in the initial turn.

CoT Length Control Figure 5 highlights the effectiveness of our turn-based advantage scaling. While GRPO suffers from severe length inflation, SOPO maintains a stable length of $\sim$1,700 tokens per turn throughout training. Crucially, this stability coincides with monotonic performance gains, proving that SOPO improves reasoning quality by eliminating verbosity bias rather than simply generating redundant "filler" tokens [17]. As detailed in Appendix D.4, while SOPO introduces a modest one-time training FLOPs overhead (+18.9%), the SOPO model reduces average token generation by 39.9% and cuts inference latency by 47.0%, yielding a highly practical framework for real-world applications.

Wider Policy Exploration. As illustrated in Figure 3, the policy entropy of SOPO remains consistently higher than that of both GRPO and CAPO throughout the training process. In terms of training dynamics, SOPO exhibits moderate entropy fluctuations even during the later stages (steps 450–600), indicating that the model retains the capacity for active exploration to further refine its performance. Conversely, GRPO and CAPO exhibit severe entropy collapse: their entropy values decay monotonically, approaching near-zero levels by step 600. This collapse indicates that the baseline policies have lost the ability to perform diverse sampling, resulting in performance plateauing where further training yields no additional gains.

5.2 Feedback Scheme

Is Socratic guidance strictly necessary? The complete Socratic feedback proposed in our method comprises two crucial components: step-wise analysis and guiding questions, which guide the student model to deduce the correct answer rather than directly revealing the ground truth. To verify the necessity of this specific design, we conducted experiments on different feedback schemes:

- **Analysis-only:** Providing step-wise analysis on the student's reasoning without the subsequent guiding question.
- **Question-only:** Providing a guiding question directly without prior step-wise analysis.

As shown in Table 3, the model trained with the original SOPO achieves the best performance compared to other methods. Notably, the "Analysis-only" method yields the lowest score, whereas the "Question-only" method achieves results comparable to SOPO. This performance gap underscores the necessity of the guiding question component. It suggests that providing heuristic questions is more effective than providing passive critiques to improve the student model to precisely locate hallucination spans.

We also conduct experiments on **SOPO with ground truth leakage**, which provides both step-wise analysis and guiding question, but allowing explicit and implicit ground truth leakage. As illustrated in Figure 6, Compared to the original SOPO, the "GT leakage" feedback results in inferior test performance, which exhibits higher rewards in the early training stage, but the original SOPO overtakes it after 120 steps. Crucially, while the final training rewards of both methods are nearly

Table 4: Performance of SOPO across diverse teacher model scales and architectures on the RAGTruth test set.

Teacher Model	F1	Prec.	Rec.
Qwen3-Next-80B-A3B-Instruct	**63.31**	**67.29**	59.77
GLM-4.7-Flash-30B-A3B	62.48	65.23	**59.96**
Qwen3-4B-Instruct-2507	62.16	66.49	58.35
Qwen3-30B-A3B-Instruct-2507	61.67	63.78	59.69
Qwen3-8B	60.47	65.51	56.14
Baseline-GRPO-4B	60.19	66.38	55.06

Table 5: Performance of GRPO and SOPO across diverse student model scales on the RAGTruth test set.

Student Model	Method	F1	Prec.	Rec.
Qwen3-1.7B	GRPO	55.8	**61.6**	51.0
	SOPO	**57.7**	61.2	**54.5**
Qwen3-4B-Thinking-2507	GRPO	60.2	66.4	55.1
	SOPO	**63.3**	**67.3**	**59.8**
Qwen3-8B	GRPO	60.6	63.5	**58.0**
	SOPO	**61.0**	**66.2**	56.5

identical (gap < 0.03), Table 3 reveals a significant performance drop (overall F1 gap > 1.0), indicating that the ground truth leakage encourages the model to overfit the RL training data, inflating the training reward scores while reducing the generalization ability. We also test **Direct-Feedback**, replacing Socratic guidance with directive, non-answer-revealing feedback (prompt in Appendix B). It yields 60.81 span-F1 versus 63.31 for SOPO (Table 3).

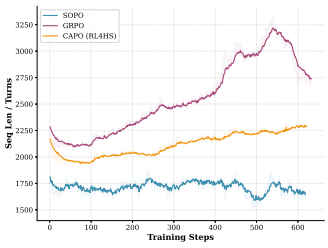

Figure 5: Sequence length per turn of RL.

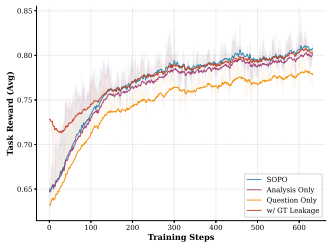

Figure 6: Mean reward of RL on different style feedback.

Table 3: RAGTruth feedback-style comparison.

Feedback Style	Span-F1	Prec.	Rec.
SOPO	**63.31**	67.29	**59.77**
w/ GT Leakage	61.65	64.43	59.09
Analysis-only	59.49	64.57	55.16
Question-only	62.25	**68.25**	57.22
Direct-Feedback	60.81	64.41	57.59

5.3 Model Scale and Heterogeneity

Teacher. We evaluate teacher models across varying scales and architectures (Table 4). While scaling to the 80B teacher maximizes Overall F1 due to a precision improvement driven by highly accurate hallucination analysis, SOPO maintains strong performance even at smaller scales. Specifically, a 4B student guided by a 4B teacher still outperforms the GRPO baseline. Notably, the newer Qwen3-2507 4B model, known to surpass legacy 8B models on complex reasoning benchmarks (e.g., AIME25, LiveBench), also yields a higher F1 score than the legacy 8B teacher. This demonstrates that intrinsic reasoning capability is more critical than number of parameters. Furthermore, a heterogeneous student-teacher configuration using a different model family (GLM-4.7-Flash, 30BA3B) provides stronger guidance than a homogeneous setup with its Qwen3-30B-A3B counterpart.

Student. To validate the robustness of student, we evaluate SOPO across diverse student capacities on RAGTruth (Table 5). SOPO consistently improves F1 across all scales, with the Qwen3-4B-2507 model yielding the largest gain (+3.1). The marginal improvements on the 1.7B and legacy 8B models stem from distinct bottlenecks: the 1.7B model lacks the long-context capacity required for long multi-turn trajectories, while the 8B model suffers from weaker intrinsic reasoning than the newer 2507 series, aligning with our teacher scaling results (Sec. 5.3).

6 Comparison with Offline Chain-of-Thought Distillation

To distinguish online teacher feedback from offline distillation, we compare SOPO with 80B-CoT-Distill. SOPO queries a frozen Qwen3-Next-80B-A3B teacher for feedback conditioned on the student's evolving reasoning during on-policy RL rollouts and updates the student using trajectory rewards. By contrast, 80B-CoT-Distill uses the same teacher to generate chain-of-thought traces offline from RAGTruth training inputs and gold hallucination lists. These traces and gold lists form

fixed supervised fine-tuning (SFT) targets for a student initialized from the same cold-start checkpoint; training uses 10,000 examples for approximately four epochs (457 steps). Thus, the baseline transfers teacher-generated reasoning through static targets rather than policy-conditioned, multi-turn feedback. Neither method uses the teacher or gold answers at inference.

Table 6: Overall RAGTruth F1 (%, mean $\pm$ standard deviation over 10 selected full-set, single-turn inference runs per checkpoint). All students share the cold-start initialization.

Method	**Training**	**Span-F1**	**Case-F1**
SOPO	Interactive RL	63.34 ± 0.22	84.65 ± 0.21
SOPO-FLOP	Interactive RL	62.96 ± 0.24	83.72 ± 0.32
80B-CoT-Distill	Teacher-trace SFT	50.09 ± 0.77	79.05 ± 0.39

As shown in Table 6, SOPO exceeds 80B-CoT-Distill by 13.25 span-F1 points (nominal paired 95% CI [10.26, 16.28]) and 5.60 case-F1 points ([3.73, 7.49]). The corresponding SOPO-FLOP differences are 12.87 ([9.77, 16.06]) and 4.67 ([2.69, 6.68]) points; all four one-sided p-values are $< 10^{-4}$. SOPO-FLOP (531 steps) is approximately FLOP-matched to GRPO (Appendix E), not to this SFT baseline.

These nominal bootstrap statistics follow Appendix E.

7 Conclusion

We propose Socratic Guided Policy Optimization (SOPO) for robust span-level hallucination detection. By integrating Socratic guidance with Turn-based Advantage Scaling, SOPO resolves reward sparsity and ensures reasoning conciseness. Our 4B model achieves state-of-the-art performance on RAGTruth and exhibits strong generalization across diverse out-of-domain benchmarks. Crucially, this enhanced reasoning efficiency substantially reduces inference latency, presenting a highly potent and cost-effective framework for aligning small language models with complex verification tasks.

8 Limitations

SOPO mitigates scalar reward instability via dense guidance, leaving advanced credit assignment techniques like integration with Process Reward Models (PRMs) for future work. Additionally, while our strict multi-turn verbosity penalty excels in verification tasks, it remains unclear whether this inadvertently suppresses the iterative trial-and-error exploration essential for open-ended generative domains like complex coding. Finally, repeated inference on fixed checkpoints provides bootstrap-based uncertainty estimates for the reported comparisons, but we have not assessed statistical significance across independent RL training seeds because full training runs are computationally costly. Nevertheless, SOPO's consistent margins across diverse benchmarks and model scales provide strong empirical confidence in its robustness.

Funding Statement

PatSnap Pte. Ltd. provided the computational resources used for this work. No other funding or third-party support was received.

Competing Interests

All authors were employed by PatSnap Pte. Ltd. during this work. The authors declare no other competing interests relevant to this work.

References

[1] Masha Belyi, Robert Friel, Shuai Shao, and Atindriyo Sanyal. Luna: An evaluation foundation model to catch language model hallucinations with high accuracy and low cost, 2024.

[2] Shreyas Chaudhari, Pranjal Aggarwal, Vishvak Murahari, Tanmay Rajpurohit, Ashwin Kalyan, Karthik Narasimhan, Ameet Deshpande, and Bruno Castro da Silva. Rlhf deciphered: A critical analysis of reinforcement learning from human feedback for llms, 2024.

[3] Muzhi Dai, Shixuan Liu, and Qingyi Si. Stable reinforcement learning for efficient reasoning, 2025.

[4] Ron Eliav, Arie Cattan, Eran Hirsch, Shahaf Bassan, Elias Stengel-Eskin, Mohit Bansal, and Ido Dagan. Clatter: Comprehensive entailment reasoning for hallucination detection, 2025.

[5] Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. Ragas: Automated evaluation of retrieval augmented generation, 2025.

[6] Mehdi Fatemi, Banafsheh Rafiee, Mingjie Tang, and Kartik Talamadupula. Concise reasoning via reinforcement learning, 2025.

[7] Yunzhen Feng, Parag Jain, Anthony Hartshorn, Yaqi Duan, and Julia Kempe. Don't waste mistakes: Leveraging negative rl-groups via confidence reweighting, 2025.

[8] Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, Tongkai Yang, Binhang Yuan, and Yi Wu. Areal: A large-scale asynchronous reinforcement learning system for language reasoning, 2025.

[9] Assaf Gerner, Netta Madvil, Nadav Barak, Alex Zaikman, Jonatan Liberman, Liron Hamra, Rotem Brazilay, Shay Tsadok, Yaron Friedman, Neal Harow, Noam Bressler, Shir Chorev, and Philip Tannor. Orion grounded in context: Retrieval-based method for hallucination detection, 2025.

[10] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020.

[11] Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. Halueval: A large-scale hallucination evaluation benchmark for large language models, 2023.

[12] Gongrui Nan, Siye Chen, Jing Huang, Mengyu Lu, Dexun Wang, Chunmei Xie, Weiqi Xiong, Xianzhou Zeng, Qixuan Zhou, Yadong Li, and Xingzhong Xu. Ngrpo: Negative-enhanced group relative policy optimization, 2025.

[13] Cheng Niu, Yuanhao Wu, Juno Zhu, Siliang Xu, Kashun Shum, Randy Zhong, Juntong Song, and Tong Zhang. Ragtruth: A hallucination corpus for developing trustworthy retrieval-augmented language models, 2024.

[14] Yuya Ogasa and Yuki Arase. Hallucinated span detection with multi-view attention features, 2025.

[15] Elisei Rykov, Kseniia Petrushina, Maksim Savkin, Valerii Olisov, Artem Vazhentsev, Kseniia Titova, Alexander Panchenko, Vasily Konovalov, and Julia Belikova. When models lie, we learn: Multilingual span-level hallucination detection with psiloqa, 2025.

[16] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024.

[17] Vaishnavi Shrivastava, Ahmed Awadallah, Vidhisha Balachandran, Shivam Garg, Harkirat Behl, and Dimitris Papailiopoulos. Sample more to think less: Group filtered policy optimization for concise reasoning, 2025.

[18] Shuzheng Si, Qingyi Wang, Haozhe Zhao, Yuzhuo Bai, Guanqiao Chen, Kangyang Luo, Gang Chen, Fanchao Qi, Minjia Zhang, Baobao Chang, and Maosong Sun. Faithlens: Detecting and explaining faithfulness hallucination, 2026.

[19] Juntong Song, Xingguang Wang, Juno Zhu, Yuanhao Wu, Xuxin Cheng, Randy Zhong, and Cheng Niu. RAG-HAT: A hallucination-aware tuning pipeline for LLM in retrieval-augmented generation. In Franck Dernoncourt, Daniel Preoţiuc-Pietro, and Anastasia Shimorina, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1548–1558, Miami, Florida, US, November 2024. Association for Computational Linguistics.

[20] Hsuan Su, Ting-Yao Hu, Hema Swetha Koppula, Kundan Krishna, Hadi Pouransari, Cheng-Yu Hsieh, Cem Koc, Joseph Yitan Cheng, Oncel Tuzel, and Raviteja Vemulapalli. Learning to reason for hallucination span detection, 2025.

[21] Zhongxiang Sun, Xiaoxue Zang, Kai Zheng, Yang Song, Jun Xu, Xiao Zhang, Weijie Yu, Yang Song, and Han Li. Redeep: Detecting hallucination in retrieval-augmented generation via mechanistic interpretability, 2025.

[22] Liyan Tang, Philippe Laban, and Greg Durrett. Minicheck: Efficient fact-checking of llms on grounding documents, 2024.

[23] Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations, 2024.

[24] Xiangfan Wu. Socratic rl: A novel framework for efficient knowledge acquisition through iterative reflection and viewpoint distillation, 2025.

[25] Jiaqi Xu, Cuiling Lan, Xuejin Chen, and Yan Lu. Stepwise think-critique: A unified framework for robust and interpretable llm reasoning, 2026.

[26] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025.

[27] Fengkai Yang, Zherui Chen, Xiaohan Wang, Xiaodong Lu, Jiajun Chai, Guojun Yin, Wei Lin, Shuai Ma, Fuzhen Zhuang, Deqing Wang, Yaodong Yang, Jianxin Li, and Yikun Ban. Your group-relative advantage is biased, 2026.

[28] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025.

[29] Xiaoying Zhang, Yipeng Zhang, Hao Sun, Kaituo Feng, Chaochao Lu, Chao Yang, and Helen Meng. Critique-grpo: Advancing llm reasoning with natural language and numerical feedback, 2026.

[30] Xingjian Zhang, Siwei Wen, Wenjun Wu, and Lei Huang. Edge-grpo: Entropy-driven grpo with guided error correction for advantage diversity, 2025.

[31] Yujia Zhou, Zheng Liu, Jiajie Jin, Jian-Yun Nie, and Zhicheng Dou. Metacognitive retrieval-augmented large language models, 2024.

[32] Ádám Kovács and Gábor Recski. Lettucedetect: A hallucination detection framework for rag applications, 2025.

A Broader Impacts

By enabling fine-grained hallucination localization, SOPO significantly enhances the safety and reliability of Retrieval-Augmented Generation (RAG) applications. This is particularly crucial in high-stakes domains, such as healthcare, legal analysis, and financial forecasting, where even minor factual errors can have severe consequences. To foster transparency and support further research in the community, we release our 1.7B and 4B models alongside the complete training codebase.

Despite these positive applications, the deployment of advanced hallucination detectors carries inherent risks. One primary concern is automation bias, where users may become overly reliant on the system and blindly trust its output, potentially missing subtle errors that the model fails to catch. Additionally, there is a risk of dual-use. Malicious actors could potentially use the precise diagnostic feedback provided by frameworks like SOPO to adversarially refine generated misinformation, making it harder for standard safety filters to detect.

To mitigate these risks, we strongly advocate that SOPO should be deployed strictly as an assistive diagnostic tool rather than an autonomous decision-maker. Maintaining a human-in-the-loop workflow is essential for final verification. Furthermore, language models inherently capture a static snapshot of information. As real-world knowledge naturally evolves and changes, the detector will require continuous updates and periodic retraining to maintain its reliability over time.

B Prompt

Prompt for RAGTruth Summarization

Below is the original document:
{reference}
Below is a summary of the document:
{response}
Your task is to determine whether the summary contains hallucinations. First, provide reasoning with the following format:
Step 1: < your first reasoning step >
Step 2: < your next reasoning step >
...(add as many steps as needed) Then, compile the labeled hallucinated spans into a JSON dict, with a key hallucination list and its value is a list of hallucinated spans. If there are potential hallucinations, the output should be in the following JSON format: hallucination list: [hallucination span1, hallucination span2, ...]. Otherwise, leave the value as an empty list as follows: hallucination list: [].

Prompt for RAGTruth Question Answering

Below is a question:
{question}
Below are the related passages:
{reference}
Below is an answer:
{response}
Your task is to determine whether the answer contains hallucinations. First, provide reasoning with the following format:
Step 1: < your first reasoning step >
Step 2: < your next reasoning step >
...(add as many steps as needed) Then, compile the labeled hallucinated spans into a JSON dict, with a key hallucination list and its value is a list of hallucinated spans. If there are potential hallucinations, the output should be in the following JSON format: hallucination list: [hallucination span1, hallucination span2, ...]. Otherwise, leave the value as an empty list as follows: hallucination list: [].

Prompt for RAGTruth Data-to-text

Below is structured data in JSON format:
{reference}
Below is an overview article written in accordance with the structured data:
{response}
Your task is to determine whether the article contains hallucinations. First, provide reasoning with the following format:
Step 1: < your first reasoning step >
Step 2: < your next reasoning step >
...(add as many steps as needed) Then, compile the labeled hallucinated spans into a JSON dict, with a key hallucination list and its value is a list of hallucinated spans. If there are potential hallucinations, the output should be in the following JSON format: hallucination list: [hallucination span1, hallucination span2, ...]. Otherwise, leave the value as an empty list as follows: hallucination list: [].

Prompt for Socratic Guidance

ROLE
You are a Socratic teacher guiding a student to self-correct their logical reasoning. You must speak directly to the student using "you" and "your".

INSTRUCTIONS
Your task is to read the student's "CoT" and compare it against the "CONTEXT" and the "CORRECT_ANSWER". Find the exact step (e.g., ## Step 2) where the student makes their first logical error or hallucination.
You must follow these strict rules:
1. NEVER reveal the "CORRECT_ANSWER" or tell the student the final True/False classification.
2. Address the student directly (e.g., "You stated that...").

HOW TO RESPOND
Write your response in exactly two natural paragraphs:
Paragraph 1 (Analysis): Directly tell the student which step contains the error. Briefly explain why their interpretation or logic in that specific step deviates from the provided CONTEXT. Do not use any special formatting or tags.
Paragraph 2 (Guidance): Ask a leading Socratic question based on the CONTEXT to help the student realize their mistake. After your question, you MUST conclude your response with this exact sentence:
"Please re-examine the logic in ## Step [n] and re-evaluate your conclusion from there." (Replace [n] with the actual step number where the error occurred).

INPUT DATA
CONTEXT
{context_fmt_str}

STATEMENT
{statement}

CORRECT_ANSWER (HIDDEN TRUTH)
{correct_answer}

CoT (Student's Chain of Thought)
{result}

Prompt for Socratic Guidance w/ Ground Truth Leakage

ROLE
You are a Corrective Logic Guide helping a student fix their reasoning errors. You must speak directly to the student using "you" and "your".

INSTRUCTIONS
Your task is to read the student's "CoT" and compare it against the "CONTEXT" and the "CORRECT_ANSWER". Find the exact step (e.g., ## Step 2) where the student makes their first logical error or hallucination.
You must follow these strict rules:
1. LIMITED SPOILERS: You ARE ALLOWED to reveal specific facts from the CONTEXT or CORRECT_ANSWER to correct a factual error in the student's current step.
2. However, you MUST NOT simply blurt out the final classification (e.g., "The answer is False") without explanation. Provide the evidence instead.
3. Address the student directly (e.g., "You stated that...").

HOW TO RESPOND
Write your response in exactly two natural paragraphs:
Paragraph 1 (Analysis): Directly tell the student which step contains the error. Explicitly contrast what they said versus what the CONTEXT actually states. It is okay to reveal the specific fact they got wrong. Do not use any special formatting or tags.
Paragraph 2 (Guidance): Provide the specific evidence or correct fact to fix their mistake. After providing the fact, you MUST conclude your response with this exact sentence:
"Please re-examine the logic in ## Step [n] using this correct information and re-evaluate your conclusion from there." (Replace [n] with the actual step number where the error occurred).

INPUT DATA
CONTEXT
{context_fmt_str}

STATEMENT
{statement}

CORRECT_ANSWER
{correct_answer}

CoT (Student's Chain of Thought)
{result}

Prompt for Direct-Feedback Ablation

ROLE
You give brief, non-Socratic, imperative feedback to a student who just failed a hallucination-detection attempt. This is an ablation baseline: feedback must be instructional, NOT analytical and NOT question-driven.

INPUT DATA
CONTEXT
{context_fmt_str}

STATEMENT
{statement}

CORRECT_ANSWER (HIDDEN TRUTH)
{correct_answer}

CoT (Student's Chain of Thought)
{result}

CRITICAL RULES (NON-NEGOTIABLE)
1. Output ONLY 1-2 short imperative sentences of direct instruction feedback.
2. Do NOT use Socratic questions (no "why...", "what if...", "can you re-check...?").
3. Do NOT write step-wise analysis, critique essays, or `<analysis>/<guidance>` tags.
4. Do NOT reveal the CORRECT_ANSWER contents, gold spans, or the final True/False label.
5. Do NOT quote long evidence from CONTEXT. Keep the hint coarse and actionable.
6. Prefer terse instructional phrasing similar to:
- "Some information is missing. Please revise your hallucination list."
- "Please check the numbers again."
- "There is a factual conflict. Re-check your conclusion."
- "A key detail is unsupported. Please inspect the statement again."
7. You may lightly adapt the instruction to the error type (missing info / numeric mismatch / conflict / unsupported claim), but stay brief and imperative.

OUTPUT
Return ONLY the short instruction text. No preamble, no markdown headings, no lists.

C Training Details

The code for our experiments is available at `https://github.com/patsnap/SOPO` and the hyper-parameters are listed in Table 7.

D Computational Cost

In this section, we provide a comprehensive breakdown of the computational costs associated with SOPO, addressing both training-time FLOPs and test-time inference latency. To ensure a strictly matched, unbiased evaluation, we compare both methods under the Dynamic Sampling (DS) configuration.

D.1 FLOPs Calculation

We estimate the total training floating-point operations (FLOPs) using standard scaling laws [10], approximating the cost as $2ND$ for forward generation and $6ND$ for backward updates, where N is the number of active parameters and D is the number of processed tokens.

Let T denote the total training trajectories and ρ denote the dynamic sampling rate. For the GRPO baseline under the DS configuration, the student model must first generate all trajectories before filtering for updates. Thus, the total FLOPs are split into full forward generation and sampled

Table 7: Hyperparameter of SFT and RL training.

Hyperparameter	**SFT (Cold-start)**	**RL**
Learning Rate	7×10^{-6}	1×10^{-6}
Batch Size	16	64
Epochs	3	4
Warmup Epoch	1	1
LR Scheduler	Cosine	Constant
Optimizer	Adam	Adam
Adam β_1	0.9	0.9
Adam β_2	0.95	0.999
Weight Decay	0.01	0.05
Group Size (G)	-	8
Rollout Temperature	-	1.0
Rollout Top-p	-	0.99
Max Tokens Per Traj	-	32768
Max Tokens Per Turn	-	8192
Max Turns (T)	-	6
Advantage Penalty Per Turn (p_{penalty})	-	0.2
Advantage Norm Level	-	Group
Feedback Reward Threshold (δ)	-	0.6
Prob Feedback (p_{feedback})	-	0.6

backward updates:

$$\mathcal{F}_{\text{GRPO}} = \underbrace{2N_{\text{stu}}D_{\text{grpo}}T}_{\text{Forward: Full } T} + \underbrace{6N_{\text{stu}}D_{\text{grpo}}(\rho \cdot T)}_{\text{Backward: Only } \rho \text{ of } T} \tag{10}$$

For SOPO, forward generation is executed for all trajectories (100% of T) using both the student (N_{stu}) and the frozen MoE teacher (N_{tea}, 3B active parameters). However, the backward update is only performed on a filtered fraction of trajectories (ρ of T) using exclusively the student model. The SOPO FLOPs formulation is:

$$\mathcal{F}_{\text{SOPO}} = \underbrace{2N_{\text{stu}}D_{\text{stu}}T + 2N_{\text{tea}}D_{\text{tea}}T}_{\text{Forward: Full } T} + \underbrace{6N_{\text{stu}}D_{\text{sopo}}(\rho \cdot T)}_{\text{Backward: Only } \rho \text{ of } T} \tag{11}$$

D.2 Empirical Cost Analysis

To provide a strictly matched computational baseline, we evaluate both models using Dynamic Sampling with $T = 320,000$.

Table 8: Trajectory statistics.

Statistic	**GRPO**	**SOPO**	**Difference**
Generation Length (multi-turn sum)	2700.1	2529.4	-6.3%
Dynamic Sampling Rate (ρ)	41.30%	48.02%	+6.72 pp
Turns Per Trajectory	1	1.57	-
Teacher Gen Length Per Feedback	-	172.8	-
Student Gen Length Per Turn	2700.1	1534.4	-43.2%

Table 9: FLOPs calculation (ExaFLOPs).

Training Phase	Component	GRPO	SOPO	Difference
Forward ($2ND$)	Student	9.37	8.62	-8.0%
	Teacher	0.00	3.46	-
Backward ($6ND$)	Student	11.61	12.87	+10.6%
Total		20.98	24.95	+18.9%

Table 10: Wall-clock time and GPU hours.

Metric	GRPO	SOPO	Difference
Training Time (Hours)	17.88	15.82	-11.5%
GPU Hours (NVIDIA-A800)	286.08	316.40	+10.6%

As detailed in Table 9, SOPO introduces an 18.9% FLOPs overhead due to the decoupled teacher forward pass. Consequently, SOPO requires 10.6% more total GPU hours (Table 10), utilizing 20 GPUs: 16 for RL training and 4 for teacher deployment. Notably, SOPO yields a student model with a 43.2% reduction in single-turn response length. This enhanced reasoning efficiency during training rollouts effectively offsets the teacher's invocation overhead, ultimately reducing the absolute wall-clock training time by 11.5%.

Furthermore, the actual inference overhead of the teacher is lower due to Automatic Prefix Caching in modern LLM inference frameworks (e.g., SGLang, vLLM). Teacher prompts contain a relatively long shared prefix, approximately 40% of the teacher's input, KV-cache is computed only once and reuses it across all rollouts.

D.3 Training Efficiency and Convergence

Beyond the reduction in generation length, the practical efficiency of SOPO is fundamentally rooted in its superior convergence rate. The dense Socratic guidance significantly reduces exploration variance, allowing SOPO to achieve the peak performance of GRPO in just 50% of the total training steps.

D.4 Test-Time Cost and Deployment Latency

The modest training overhead translates into massive efficiency gains during deployment. As shown in Table 11 and Figure 7, the SOPO student model naturally suppresses verbosity, reducing average token counts by 39.9% and cutting inference latency by 47.0%. Trading a one-time 18.9% training overhead for a 47% reduction in real-world deployment latency represents a highly practical and significant contribution to the community.

Table 11: Average inference latency and token counts during testing on RAGTruth (Concurrency = 32).

Method	Latency Avg (ms)	Latency Std	# Token Avg
GRPO	478.2	225.1	1250.2
SOPO	**253.6 (-47.0%)**	81.8	**750.9 (-39.9%)**

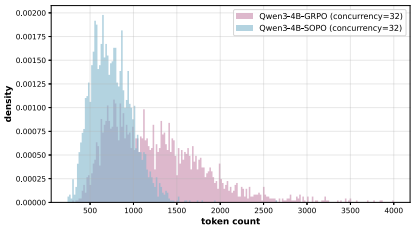

Figure 7: Output Length Distributions of SOPO and GRPO on RAGTruth.

Table 12: Repeated-inference RAGTruth results (%, 10 runs per method). Nominal paired-bootstrap 95% CIs and one-sided p-values use 10,000 replicates (seed 42), drawing one run per pool and resampling matched test examples. Estimates are conditional on these pools and do not measure training-seed variability.

SOPO checkpoint	Baseline	SOPO mean	Base mean	Δ F1	95% CI for Δ	One-sided p
Overall span-F1						
SOPO	CAPO	63.34	59.75	+3.59	[1.52, 5.75]	0.0005
	GRPO		59.49	+3.85	[1.33, 6.57]	0.0008
SOPO-FLOP	CAPO	62.96	59.75	+3.21	[0.98, 5.38]	0.0031
	GRPO		59.49	+3.47	[0.93, 6.00]	0.0039
SOPO-GPUh	CAPO	62.74	59.75	+2.99	[0.95, 5.07]	0.0016
	GRPO		59.49	+3.25	[0.80, 5.60]	0.0057
Overall case-F1						
SOPO	CAPO	84.65	83.03	+1.61	[0.07, 3.07]	0.0209
	GRPO		83.03	+1.61	[0.00, 3.25]	0.0248
SOPO-FLOP	CAPO	83.72	83.03	+0.69	[-1.00, 2.37]	0.2189
	GRPO		83.03	+0.69	[-1.07, 2.44]	0.2293
SOPO-GPUh	CAPO	84.07	83.03	+1.04	[-0.56, 2.64]	0.0990
	GRPO		83.03	+1.04	[-0.60, 2.75]	0.1138

E Inference-Time Significance on RAGTruth

Compute-matched checkpoints. Let $S = 632$ be the full-training step count for both GRPO and SOPO. Under the dynamic-sampling configuration in Appendix D, let F_m and H_m denote the full-budget FLOPs and GPU-hours for method m, respectively. Table 9 gives $(F_{\mathrm{GRPO}}, F_{\mathrm{SOPO}}) = (20.98, 24.95)$ ExaFLOPs for $T = 320{,}000$ trajectories, and Table 10 gives $(H_{\mathrm{GRPO}}, H_{\mathrm{SOPO}}) = (286.08, 316.40)$ GPU-hours. Approximating the SOPO cost per step by F_{SOPO}/S or H_{SOPO}/S, we choose the nearest integer step counts that match GRPO's full-budget costs:

$$s_{\mathrm{FLOP}} = \mathrm{round}\left(\frac{F_{\mathrm{GRPO}}}{F_{\mathrm{SOPO}}/S}\right) = \mathrm{round}\left(\frac{632 \times 20.98}{24.95}\right) = 531, \tag{12}$$

$$s_{\mathrm{GPUh}} = \mathrm{round}\left(\frac{H_{\mathrm{GRPO}}}{H_{\mathrm{SOPO}}/S}\right) = \mathrm{round}\left(\frac{632 \times 286.08}{316.40}\right) = 571. \tag{13}$$

Equation (12) defines SOPO-FLOP, and Eq. (13) defines SOPO-GPUh. Using average per-step costs, these checkpoints approximately match GRPO's 632-step FLOP and GPU-hour budgets, respectively.

For each checkpoint and baseline, we evaluate 10 full-set inference runs on the fixed 2,700-example RAGTruth test set. Tables 1 and 2 report single-run scores; Table 12 summarizes the repeated-inference results. Differences are computed before rounding the displayed means.

At the checkpoints approximately matched to GRPO's FLOP and GPU-hour budgets, SOPO improves overall span-F1 over GRPO by 3.47 and 3.25 points, respectively, and over CAPO by 3.21 and 2.99 points. All four nominal paired 95% bootstrap intervals exclude zero (one-sided $p \leq 0.0057$; Table 12).

F Evaluation on Additional Benchmarks

We evaluate the full-training SOPO checkpoint and its FLOP-matched checkpoint (SOPO-FLOP, 531 steps) on four out-of-domain (OOD) benchmarks using their primary metrics: span-F1 for PsiloQA and Math-Shepherd-614, the unweighted mean balanced accuracy (BAcc) for LLM-AggreFact, and overall accuracy for HaluEval. Table 13 presents means and standard deviations across 10 full-set inference runs per method; Table 14 presents nominal run-pool and sample-paired bootstrap

Table 13: OOD primary-metric means $\pm$ standard deviations (%) over 10 inference runs per checkpoint or baseline group.

Checkpoint	PsiloQA span-F1	Math-Shepherd span-F1	AggreFact mean BAcc	HaluEval accuracy
SOPO	74.07 ± 0.28	89.13 ± 0.36	75.98 ± 0.11	69.78 ± 0.02
SOPO-FLOP	74.43 ± 0.24	88.00 ± 0.46	75.27 ± 0.08	69.25 ± 0.05
CAPO	72.86 ± 0.30	86.98 ± 1.04	75.31 ± 0.14	69.40 ± 0.05
GRPO	72.71 ± 0.30	86.19 ± 0.91	73.64 ± 0.31	67.89 ± 0.07

Table 14: OOD primary-metric run-pool and sample-paired bootstrap comparisons (percentage points). Point estimates are differences of unrounded means over 10 inference runs per method. Nominal 95% CIs and one-sided p-values use $B = 10{,}000$ replicates (seed 42), drawing one run per pool and resampling matched test examples; these estimates are conditional on the run pools.

Benchmark	SOPO checkpoint	Baseline	Δ	95% CI for Δ	One-sided p
PsiloQA	SOPO	CAPO	+1.21	$[-0.45, 2.89]$	0.0789
		GRPO	+1.36	$[-0.41, 3.31]$	0.0708
	SOPO-FLOP	CAPO	+1.58	$[0.01, 3.30]$	0.0248
		GRPO	+1.73	$[0.15, 3.54]$	0.0156
Math-Shepherd	SOPO	CAPO	+2.16	$[-1.04, 5.50]$	0.0917
		GRPO	+2.94	$[0.10, 6.24]$	0.0204
	SOPO-FLOP	CAPO	+1.02	$[-2.50, 4.60]$	0.2757
		GRPO	+1.81	$[-1.48, 5.49]$	0.1553
LLM-AggreFact	SOPO	CAPO	+0.68	$[-0.26, 1.59]$	0.0774
		GRPO	+2.35	$[1.28, 3.42]$	$< 10^{-4}$
	SOPO-FLOP	CAPO	-0.04	$[-1.03, 0.91]$	0.5293
		GRPO	+1.63	$[0.52, 2.69]$	0.0022
HaluEval	SOPO	CAPO	+0.38	$[0.07, 0.68]$	0.0083
		GRPO	+1.89	$[1.56, 2.24]$	$< 10^{-4}$
	SOPO-FLOP	CAPO	-0.15	$[-0.48, 0.18]$	0.8220
		GRPO	+1.37	$[1.03, 1.72]$	$< 10^{-4}$

comparisons. The reported point estimate is the difference between the methods' unrounded 10-run mean primary scores. For each bootstrap replicate, we draw one run from each method's 10-run pool and resample matched test examples with replacement, recomputing the difference on those examples. Thus all 10 runs contribute to the resampling pool; the intervals are conditional on these pools and do not measure variability across independent training seeds.

F.1 Span-Level: PsiloQA

PsiloQA [15] evaluates fine-grained QA on its English subset using span-F1. SOPO averages 74.07 over 10 inference runs, compared with 72.86 for CAPO and 72.71 for GRPO; SOPO-FLOP averages 74.43 (Table 13). The full SOPO checkpoint's nominal 95% bootstrap intervals cross zero against both baselines, whereas SOPO-FLOP's intervals are positive (Table 14).

F.2 Case-Level: LLM-AggreFact

LLM-AggreFact [22] aggregates grounded factuality datasets; we exclude its RAGTruth subset and report the unweighted mean balanced accuracy across the remaining datasets. SOPO averages 75.98 versus 75.31 for CAPO and 73.64 for GRPO. Its nominal 95% bootstrap interval versus GRPO is positive, while the interval versus CAPO crosses zero (Table 14).

F.3 Case-Level: HaluEval

On HaluEval [11], we compare overall binary accuracy. SOPO averages 69.78 over 10 inference runs, versus 69.40 for CAPO and 67.89 for GRPO; its nominal 95% bootstrap intervals are above zero against both baselines (Table 14).

F.4 Out-of-Domain: Math-Shepherd

To construct Math-Shepherd-614, we randomly sample 1,000 instances from the public Math-Shepherd stepwise-supervision split [23] (seed 42). Each original trajectory (problem and reasoning steps) is sent unchanged to Claude-4.6-Sonnet, which independently judges the correctness of each step given the problem and preceding steps, producing a per-step true/false list. We retain an instance only when every judgment agrees with its original Math-Shepherd step label; otherwise we discard it, without rewriting any labels. This noise-filtering procedure retains 614 instances and discards 386 (38.6%). The retained set contains 1,794 correct and 1,395 incorrect steps; 334 instances have at least one incorrect step, and 280 are all-correct. On this fixed set, SOPO averages 89.13 span-F1, compared with 86.98 for CAPO and 86.19 for GRPO; its nominal 95% bootstrap interval versus GRPO is above zero, while the interval versus CAPO crosses zero (Table 14).

G Analysis of Hard Case on RAGTruth Test Set

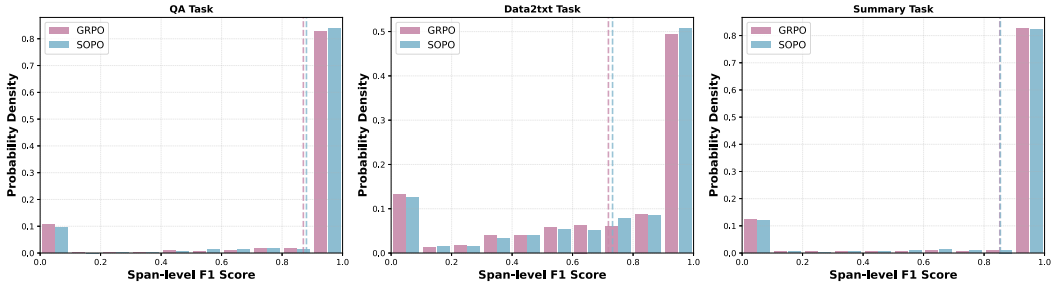

Figure 8: Probability density histogram of QA, data2text and summary task on RAGTruth Test set.

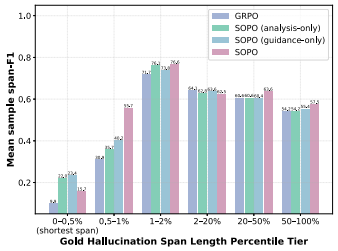

Figure 9: Mean sample span-F1 across Gold Hallucination Span Length Percentile Tiers.

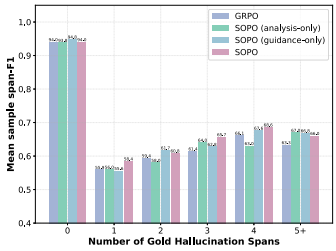

Figure 10: Mean sample span-F1 across the Number of Gold Hallucination Spans.

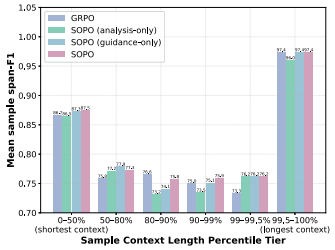

Figure 11: Mean sample span-F1 across Sample Context Length Percentile Tiers.

Breakthrough on Persistent Errors. We conducted a fine-grained analysis of span-level F1 scores across all 2,700 samples covering three subtasks in the RAGTruth test set. As illustrated in Figure 8, SOPO exhibits a significantly lower probability density in the extremely low-score interval $(0.0 - 0.1)$ compared to standard GRPO. This trend extends into the lower-mid range $(0.1 - 0.5)$, highlighting a breakthrough in handling hard examples: SOPO effectively minimizes the rate of complete localization failures, converting zero-score predictions into valid detections, and driving a positive distribution shift toward higher scores. Notably, in the high-score interval $(0.8 - 1.0)$, SOPO consistently surpasses GRPO, particularly in the QA and Data-to-Text subtasks. These results confirm that SOPO achieves superior span-level localization precision.

To provide direct evidence beyond macroscopic score histograms, we evaluated SOPO on specific "Hard Examples" categorized along three dimensions known to exacerbate reward sparsity and localization ambiguity:

- **Samples with Short Spans (Bottom 2% in length):** Highly sensitive cases where even slight boundary deviations drastically penalize the span-F1 score.

- **Samples with Multiple Spans ($\geq$3 spans):** Dense cases requiring the exhaustive detection of scattered hallucinations.
- **Samples with Long Contexts (Top 10% in length):** Information-heavy cases that challenge the model's ability to filter massive amounts of noisy context.

G.1 Performance on Short Spans

Numerical Errors (0–0.5% length). As shown in Figure 9, when a hallucination involves merely a mismatched number or date, the task reduces to pure string matching rather than logical deduction; consequently, the full Socratic chain offers limited marginal benefit.

Logical Phrases (0.5–1% length). For subtle logical errors, however, full SOPO proves critical. While the standard GRPO collapses to an F1 of 0.309 and the "Guidance-only" variant plateaus at 0.402, full SOPO achieves 0.557 (a 38.5% relative improvement over Guidance-only). This proves that simply asking a heuristic question leaves the student guessing; explicit diagnostic reasoning ("Analysis") is strictly required to anchor precise localization.

G.2 Performance on dense and sparse spans

In Figure 10, counterintuitively, our analysis reveals that the most persistent "hard samples" are often those containing exactly one hallucinated span, as the model struggles to suppress false positives in otherwise factual text. In this sparse setting, the "Guidance-only" variant (0.556 F1) actually underperforms standard GRPO (0.559 F1), demonstrating that heuristic questioning can induce over-correction without explicit diagnostic anchoring. Full SOPO (0.584 F1) successfully overcomes this limitation via its systematic "Analysis", achieving the highest performance in both sparse (1 span) and highly dense ($\geq$4 spans) scenarios.

G.3 Performance on Long Context

In Figure 11, while detection performance generally degrades as context length increases, the difficulty curve is not perfectly linear. Across the most challenging length percentiles (50%–99.5%), full SOPO consistently outperforms GRPO and all ablated variants, demonstrating robust noise-filtering capabilities and sustained attention over extended reasoning trajectories.

H Advantage Scaling

Pitfalls of Heuristic Post-Normalization Advantage Scaling. GRPO relies on group-wise normalization to ensure that the mean advantage within each group is zero. This property acts as a dynamic baseline, minimizing gradient variance. While recent research suggests that scaling normalized advantages is feasible under strict constraints to correct the group-relative advantage estimator bias [27], applying arbitrary heuristic scaling after normalization (such as class-aware advantage scaling from CAPO) is mathematically unsound.

Specifically, if the scaling factor is correlated with the advantage (e.g., penalizing longer chains which might have higher/lower rewards), the weighted sum of the final advantages diverges from zero:

$$\mathbb{E}[A'_{\mathrm{i}}] \neq 0 \quad \text{where} \quad A'_{\mathrm{i}} = \alpha_i \cdot A_{\text{norm}} \tag{14}$$

This reintroduces a baseline bias into the gradient estimator. Unlike theoretically derived corrections that aim to restore unbiasedness, heuristic post-norm scaling arbitrarily distorts the optimization objective, leading to training instability. As illustrated in Figure 2, SOPO with post-normalization advantage scaling exhibits severe instability around step 100, accompanied by a sharp decline in reward.

Rationale for Pre-Normalization Intervention. In contrast, our method applies the scaling factor α_i directly to the raw reward *prior to* normalization:

$$A_{\mathrm{i}} = \text{Normalize}(\alpha_i \cdot R_{\mathrm{i}}) \tag{15}$$

This distinction is crucial: we intervene at the valuation level rather than the estimator level. By reshaping the reward before normalization, we effectively lower the relative rank of verbose responses within the group. The subsequent normalization step then naturally handles the distribution, ensuring that the final advantages remain centered at zero with unit variance. This approach effectively incorporates our heuristic constraints without violating the mathematical integrity of the advantage estimator.

I Calibration of RL4HS Scores

The discrepancy between our "Overall" scores (Table 1) and the "Average" scores reported in the original RL4HS [20] paper arises from two specific deviations from the standard RAGTruth evaluation protocol:

Aggregation Mismatch (Macro vs. Micro-averaging). RL4HS reported a simple macro-average across tasks. In this approach, the precision (P_i), recall (R_i), and F1 score ($F1_i$) are computed for each task independently, and the final score is the arithmetic mean of these individual F1 scores. Let N be the total number of tasks. The macro-averaged F1 is formally defined and expanded as:

$$F1_{macro} = \frac{1}{N}\sum_{i=1}^{N} F1_i = \frac{1}{N}\sum_{i=1}^{N} \frac{2 \cdot P_i \cdot R_i}{P_i + R_i} \tag{16}$$

In contrast, the official benchmark and all standard baselines (e.g., LettuceDetect [32], RAG-HAT [19], Ragas [5]) strictly employ a global micro-average. This protocol aggregates the raw true positives (TP), false positives (FP), and false negatives (FN) across all tasks before computing the final metrics:

$$\begin{aligned} P_{micro} &= \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N}(TP_i + FP_i)}, \quad R_{micro} = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N}(TP_i + FN_i)} \\ F1_{micro} &= \frac{2 \cdot P_{micro} \cdot R_{micro}}{P_{micro} + R_{micro}} \end{aligned} \tag{17}$$

Because the subtasks in RAGTruth contain significantly different numbers of instances and hallucination densities, macro-averaging disproportionately weights smaller tasks, leading to skewed overall results.

Test Set Mismatch. Furthermore, according to Appendix A.5 (Dataset Statistics) of the RL4HS paper, their QA task evaluation utilized only 875 test instances (160 + 715), deviating from the official 900 instances provided by RAGTruth.

To ensure an unbiased comparison, we standardized their results. We counted the ground-truth spans for each task in the official test set and scaled the QA subset by a factor of 875/900 to match their data distribution. Using their reported per-task metrics, we then mathematically reverse-engineered the global confusion matrix (deriving the exact TP, FP, and FN counts for each task) to compute the correct micro-averaged Overall F1.

J Synergy of the Dual-Penalty Mechanism for Verbosity Control

In multi-turn reinforcement learning, models often exploit reward hacking to maximize scores without genuinely solving the task. To cultivate an autonomous and accurate student policy, SOPO employs two complementary mechanisms to prevent different hacking strategies and strictly control reasoning verbosity.

Turn-Averaged Reward (Local Quality). If trained solely on the final-turn reward, the model could intentionally generate an empty or poor initial response merely to solicit the teacher's hints. By averaging span-level F1 scores across all turns, any early failure permanently degrades the overall score. This strict local quality constraint compels the model to provide its best reasoning attempt at every single step.

Advantage Scaling (Group-Relative Efficiency). The averaged reward alone cannot distinguish between trajectories with different interaction lengths but identical average scores. For example, consider a two-turn trajectory where the student scores 0.6 in the first turn and 0.7 in the second, resulting in a mean reward of 0.65. Without further intervention, this is indistinguishable from a more efficient single-turn trajectory that directly achieves a score of 0.65. Standard GRPO would assign these identical advantages, inadvertently encouraging interaction redundancy. Advantage Scaling directly penalizes the two-turn case by applying a scaling factor before group normalization. This intra-group efficiency constraint explicitly signals to the optimizer that shorter, highly autonomous trajectories hold a superior relative advantage.

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: See Sections 3, 5.

   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 explicitly discuss the limitations of our work, including potential challenges with credit assignment and open-ended generative exploration, in Section 8.

   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: This paper does not have theoretical results.

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 provide comprehensive details regarding experimental setups, datasets, and prompts in Section 4, Appendix B and C to ensure the reproducibility of our main claims.

   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: We provide training data, code, and evaluation resources via the SOPO repository at `https://github.com/patsnap/SOPO`.

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: Full experimental details, including dataset splits, model configurations, and a complete hyperparameter table, are provided in Section 4 and Appendix 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: Appendices E and F report means and standard deviations over repeated inference runs, together with nominal paired-bootstrap confidence intervals and one-sided p-values. Their captions specify the resampling procedures and the inference-run pools on which the intervals are conditional; independent training-seed variability is not assessed.

   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 provide a detailed breakdown of the required computational resources, including hardware types, GPU hours, and FLOPs calculations, in Appendix D.

   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: The research conducted and the public datasets utilized in this paper fully comply with the NeurIPS Code of Ethics.

   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: Both the potential positive applications in safer RAG systems and negative societal impacts are discussed in Section A.

    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: [N/A]

    Justification: The paper does not pose such risks.

    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: The RAGTruth corpus is cited and its official repository specifies the MIT License (`https://github.com/ParticleMedia/RAGTruth`). The Qwen3-4B-Thinking-2507 student and Qwen3-Next-80B-A3B-Instruct teacher models are released under Apache License 2.0 by Qwen. Our training and evaluation code is publicly released under Apache License 2.0 at `https://github.com/patsnap/SOPO`, which also links our released SOPO model weights and training data. Third-party assets and API services retain their original licenses and terms of use; the SOPO code license does not override them.

    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: The new assets introduced in this work, including the SOPO training data, code and model weights, are well-documented in the SOPO repository at `https://github.com/patsnap/SOPO`.

    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 paper does not involve any crowdsourcing experiments or 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: [N/A]

Justification: This paper does not involve human subject research, therefore Institutional Review Board (IRB) approval is not applicable.

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: [N/A]

Justification: The present research does not involve LLMs as an important component.

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.