Think More, Hallucinate Less: Mitigating Hallucinations via Dual Process of Fast and Slow Thinking

Xiaoxue Cheng[1*], **Junyi Li**[2*], **Wayne Xin Zhao**[1†] and **Ji-Rong Wen**[1]
[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]Department of Computer Science, National University of Singapore
chengxiaoxue@ruc.edu.cn junyi_cs@nus.edu.sg batmanfly@gmail.com

Abstract

Large language models (LLMs) demonstrate exceptional capabilities, yet still face the hallucination issue. Typical text generation approaches adopt an auto-regressive generation without deliberate reasoning, often leading to untrustworthy and factually inaccurate responses. In this paper, we propose **HaluSearch**, a novel framework that incorporates tree search-based algorithms (*e.g.,* MCTS) to enable an explicit slow thinking generation process for mitigating hallucinations during inference. Specifically, HaluSearch frames text generation as a step-by-step reasoning process, using a self-evaluation reward model to score each generation step and guide the tree search towards the most reliable generation pathway. To balance efficiency and quality, we introduce a hierarchical system switch mechanism inspired by the dual process theory in cognitive science , which dynamically switches between fast and slow thinking modes at both instance and step levels. We conduct extensive experiments on both English and Chinese datasets, and the results show that our approach significantly outperforms baseline approaches.

1 Introduction

Large language models (LLMs) (Zhao et al., 2023) are revolutionizing the landscape of artificial intelligence, showcasing remarkable capabilities in generating human-quality text and tackling diverse language tasks. Despite these advancements, they often struggle with the issue of hallucination (Ji et al., 2023; Huang et al., 2023; Rawte et al., 2023; Ye et al., 2023; Zhang et al., 2023), where responses can be untrustworthy or factually inaccurate. This issue significantly impacts the practical applications of LLMs in real-world scenarios. Existing studies (Xu et al., 2024; Banerjee et al., 2024) indicate that due to limitations in training data, model architecture, training method, and other factors, completely eliminating hallucinations is infeasible. Therefore, the development of effective techniques to mitigate hallucinations is critical for improving the reliability and robustness of LLM outputs.

Existing efforts to mitigate hallucinations have targeted different stages of the LLM pipeline, including pre-training (Li et al., 2023b), supervised fine-tuning (Tian et al., 2024; Elaraby et al., 2023; Lin et al., 2024), and inference (Dhuliawala et al., 2024; Madaan et al., 2024; Kang et al., 2023). In this study, we mainly focus on the hallucination mitigation techniques during the inference stage. Existing approaches can be broadly divided into two categories, *i.e.,* retrieval-augmented generation (RAG) and internal knowledge-based methods. RAG methods (Lewis et al., 2020) enhance response accuracy by retrieving documents relevant to the query and incorporating them as additional contextual information (Asai et al., 2024). Internal knowledge-based methods, such as step-by-step reasoning (Wei et al., 2022), self-verification (Dhuliawala et al., 2024), and self-consistency (Wang et al., 2023), rely on instructions to generate intermediate reasoning steps or utilize the model's consistency by selecting the most coherent response from multiple outputs. Although these studies perform deliberative reasoning to mitigate hallucinations, they operate at the response level and remain constrained by the auto-regressive generation paradigm, where intermediate errors can accumulate, potentially leading to incorrect final outputs.

In this paper, we propose **HaluSearch**, a novel framework that explicitly models response generation as a deliberate thinking process of System 2 (Kahneman, 2011), incorporating a dynamic system switch mechanism to adaptively alternate between fast and slow thinking modes. To achieve this goal, we first integrate tree search-based algorithms (*e.g.,* MCTS) to formulate text generation as a step-by-step reasoning process, treating each

* Equal contributions
† Corresponding author

Findings of the Association for Computational Linguistics: ACL 2025, pages 7979–7990
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

sentence as an individual reasoning step. Secondly, inspired by the interaction between System 1 and System 2 in the dual process theory of cognitive science (Wason and Evans, 1974), we propose a dynamic system switch mechanism at both instance and step levels within the text generation process. Starting with the input prompt, the dynamic system switch mechanism is employed to determine the appropriate thinking mode for the input question or each reasoning step: fast thinking directly generates a completed response or a single reasoning step, while slow thinking generates multiple intermediate sentences that are evaluated by a reward model. Finally, we employ the HaluSearch framework to synthesize preference data to train a reward model for assessing the degree of hallucinations in generated sentences. Given the challenges in training an accurate reward model, we explore two approaches: generative reward modeling and critique-based reward modeling, targeting effective self-evaluation to guide the search process. Compared to previous work, our approach performs step-level reasoning to generate responses rather than relying on response-level refinements, which can achieve more effective and fine-grained hallucination mitigation.

We conduct extensive experiments to evaluate the effectiveness of HaluSearch using Llama3.1-8B-Instruct and Qwen2-7B-Instruct as policy models. The results show that our method achieves substantial improvements over previous prompt-based and inference-time intervention baselines across both English and Chinese datasets.

2 Related Work

2.1 Halluciantion Mitigation

Studies on hallucination mitigation span both the training and inference stages of LLMs (Li et al., 2024a). Due to the high computational cost and resource requirements of model training, more research has focused on exploring hallucination mitigation methods during the inference stage, which can be broadly divided into two categories. The first category is retrieval-augmented generation (RAG) (Li et al., 2024a) which reduces hallucinations by retrieving documents relevant to the query and providing them as additional context, relying on external knowledge to improve response accuracy. Another category of methods, including Chain-of-Thought (CoT) (Wei et al., 2022), Self-Consistency (Wang et al., 2023), and Best-of-N (Lightman et al., 2024), seeks to mitigate hallucinations by leveraging the internal knowledge of LLMs through prompt-based reasoning or consistency-driven strategies. However, these approaches operate at the response level and are still constrained to the fast and intuitive auto-regressive generation paradigm. In contrast, our approach employs MCTS to frame response generation as an explicit step-by-step slow thinking process, utilizing step-level rewards to explore optimal reasoning paths and generate more reliable responses.

2.2 System 2 Thinking in LLMs

System 2 thinking in LLMs emulates the human process of deliberate reasoning to generate high-quality and accurate responses. Many studies (Snell et al., 2024; Min et al., 2024; Tang et al., 2025) have demonstrated that scaling inference-time computation serves as an alternative to training for enhancing the performance of LLMs, particularly in complex reasoning scenarios (*e.g.,* mathematical problem solving). Early works primarily implemented System 2 thinking in LLMs by using prompts to guide the generation of intermediate reasoning steps, such as CoT (Wei et al., 2022), ToT (Yao et al., 2024). Recent approaches (Wang et al., 2024; Kang et al., 2024; Jiang et al., 2024; Tang et al., 2024) focus on incorporating search-augmented reasoning during the decoding process to explicitly implement System 2 thinking, which have shown considerable gains. However, the potential of this approach for hallucination mitigation has not been fully investigated. In this paper, we investigate whether tree search-based slow thinking can effectively leverage accurate internal knowledge from LLMs to mitigate hallucinations.

3 Approach

In this paper, we propose **HaluSearch**, leveraging deliberate planning to mitigate LLM hallucinations during the inference stage. Previous work on mitigating hallucinations during inference is mainly limited to the fast thinking paradigm that relies on prompts to instruct LLMs to generate faithful responses or directly calibrates the internal generation mechanism of LLMs (Dhuliawala et al., 2024; Madaan et al., 2024; Li et al., 2024b). However, these approaches have not fully exploited the internal knowledge of LLMs to address hallucinations. Some work (Wang et al., 2023; Orgad et al., 2024) found that hallucinations often arise from

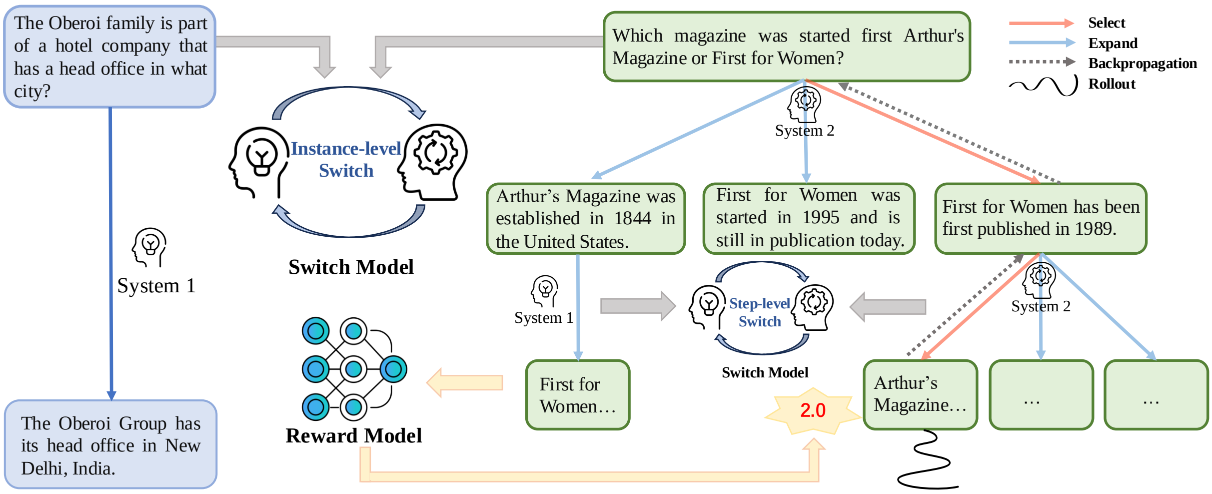

Figure 1: The overview of our proposed HaluSearch approach. The left part demonstrates the process of fast thinking response generation (System 1). The right part illustrates the tree search process with step-level switch between fast thinking and slow thinking generation (System 2).

ineffective generation, even when the model possesses knowledge of the underlying facts. Drawing inspiration from the success of tree search-based algorithms in complex reasoning (Kang et al., 2024; Wang et al., 2024; Jiang et al., 2024), we propose to integrate Monte Carlo Tree Search (MCTS) with a system switch mechanism between fast and slow thinking modes. Our approach enables an explicit deliberate planning process to fully exploit the internal knowledge of LLMs for reducing hallucinations. The overall architecture of HaluSearch is shown in Figure 1.

3.1 Problem Formulation

Our proposed approach aims to mitigate hallucinations in the inference stage. Formally, given an input prompt x, the LLM is instructed to generate a response $y = \langle y_1, ..., y_t, ..., y_T \rangle$, where y_t denotes the t-th sentence. Specifically, we formulate the response generation process of LLMs as step-by-step reasoning, where each sentence corresponds to an intermediate reasoning step. In our approach, MCTS aims to construct a search tree $\mathcal{T}$ based on the target LLM. In this tree, a node in the t-th tree level is represented as $s_t = \{y_t, N(s_t), V(s_t)\}$, where y_t refers to the generated sentence, $N(s_t)$ denotes the visit count, and $V(s_t)$ represents the value score. The root node $s_0 = \{x\}$ contains only the initial input prompt. The final response is the concatenation of sentences $\langle y_1, ..., y_T \rangle$ where each sentence y_t comes from a node on the path from the root node to the leaf node. The target LLM is referred to as the policy model π_θ, and the reward model is denoted as R.

3.2 Monte Carlo Tree Search

In our approach, the MCTS-driven generation process operates as an iterative procedure, where each iteration consists of four key steps: selection, expansion, evaluation, and backpropagation. Specifically, the MCTS process begins by initializing the root node of the tree s_0 with the input prompt.

Selection. The selection process starts from the root node s_0 and selects the leaf node with the highest exploration potential, determined by the UCT (Upper Confidence Bounds applied to Trees) (Kocsis and Szepesvári, 2006) score. The UTC score is calculated as follows:

$$UCT(s_t) = V(s_t) + w\sqrt{\frac{\ln N(p)}{N(s_t)}}, \qquad (1)$$

where w is a hyper-parameter that balances the exploitation (*i.e.,* node value $V(s_t)$) and exploration (*i.e.,* visit count $N(s_t)$), and p denotes the parent node of s_t.

Expansion. After selecting the node with the highest UCT score, it is expanded by generating multiple child nodes. Based on the historical information, the policy model is employed to generate the next sentence as follows:

$$y_{t+1} \sim \pi_\theta(\cdot|x, \{y_i\}_{i=1}^t), \qquad (2)$$

where the previously generated sentences $\{y_i\}_{i=1}^t$ is regarded as the historical context, and the policy model samples and generates K sentences y_{t+1} as a set of child nodes $\mathcal{C}(s_{t+1})$. Compared to previous work (Lin et al., 2024; Xie et al., 2024), which mainly focused on a fast thinking generation paradigm, our approach leverages MCTS to explore multiple potential generations for fully exploiting the internal knowledge of LLMs.

Evaluation. Each expanded node is evaluated to obtain its value $V(s_{t+1})$. Specifically, the policy model performs rollout to complete the state of each child node s_{t+1} by sampling m completed responses. These responses, denoted as $\mathcal{C}_{\mathrm{r}}(s_{t+1})$, are then evaluated by the reward model (Section 3.3), which assigns a reward score to each response. The average score r of the m completed responses is used as the initial value $V(s_{t+1})$ of the corresponding child node s_{t+1}.

Backpropagation. After evaluating the expanded nodes, their values are backpropagated along the traversal path to update the visit counts and value scores of the ancestor nodes s_j $(0 \leq j \leq t)$. The updates are performed as follows:

$$N_{\text{new}}(s_j) = N_{\text{old}}(s_j) + 1, \tag{3}$$

$$V_{\text{new}}(s_j) = \frac{V_{\text{old}}(s_j)N_{\text{old}}(s_j) + r}{N_{\text{new}}(s_j)}, \tag{4}$$

where $N_{\text{old}}(s_j)$ and $V_{\text{old}}(s_j)$ represent the last visit count and value score of node s_j before backpropagation, respectively, and r is the reward obtained from the evaluation step.

The above four stages are performed iteratively until the policy model reaches the terminal state. We define two termination conditions for MCTS as follows:

1. The maximum MCTS iteration M is reached.

2. A terminal node is encountered where the reward satisfies the reward threshold, indicating a lower likelihood of hallucinations.

Once the tree search is completed, the optimal path from the root node to the terminal node is selected using a greedy strategy that prioritizes nodes with the highest value scores and their associated sentences are combined as the final response.

3.3 Self-Evaluation Reward Model

In the MCTS-driven generation framework, the reward model plays a crucial role in evaluating the child nodes at each step and guiding the tree search towards more promising directions. While a straightforward approach involves using an advanced LLM (*e.g.,* GPT-4) as the reward model, this approach heavily depends on closed-source LLMs. To enable effective self-evaluation, we train the reward model on the same foundation model as the policy model, exploring two reward modeling approaches: generative reward modeling and critique-based reward modeling.

Training Data. We collect existing question-answering datasets on hallucination evaluation and sample a subset of questions to construct the training data for the reward model. To obtain diverse reward data, we adopt the MCTS generation framework described in Section 3.2 to generate the response-score pairs. For generative reward data, we first collect all complete responses through rollouts and then employ GPT-4 to give a reward score to these completed responses, following a Likert scale-based approach (Likert, 1932) ranging from 1 to 5. Higher scores indicate greater hallucinations, with explicit criteria provided for each score level. To enhance scoring reliability, the ground truth answer for each question is included in the prompt as a reference. For critique-based reward data, GPT-4 first generates a detailed critique of the response about its correctness and then gives a reward score based on the critique. This format allows the reward model to incorporate critiques as the rationale in its scoring process, facilitating a more accurate evaluation of responses. To create a balanced training dataset for the reward model, these collected responses are deduplicated based on TF-IDF cosine similarity and the samples are adjusted to ensure an uniform distribution on score levels. Detailed scoring guidelines and instructions are provided in the Appendix B.

Training Method. Leveraging the generative capabilities of LLMs, we utilize the data collected above to train the reward model in a supervised fine-tuning manner. Specifically, for generative reward modeling, the reward model takes the response y as input and directly predicts the corresponding reward score r. For critique-based reward modeling, the model is trained to first generate the critique c and then predict the reward score r. The training objective is defined by the cross-entropy loss:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\begin{cases} \log P(r^{(i)} \mid y^{(i)};\theta), & \text{if no critique,} \\ \log P(c^{(i)}, r^{(i)} \mid y^{(i)};\theta), & \text{if critique,} \end{cases}$$

where N is the number of training samples.

3.4 Dynamic System Switch

In existing research (Kahneman, 2011; Jiang et al., 2024; Yu et al., 2024), search-based decoding approaches that improve response quality by scaling inference time are referred to as System 2 thinking mode, whereas direct generation methods are termed System 1 thinking mode (Kahneman, 2011). While System 2 provides superior response quality, its substantially computational costs make it impractical for universal application, as not all user queries necessitate complex reasoning. To balance efficiency and response quality, we propose a dynamic system switch mechanism that adaptively selects the appropriate thinking mode. The mechanism operates on two hierarchical levels: *instance-level switch*, which determines the generation mode for the input question, and *step-level switch*, which adjusts the generation mode for individual reasoning steps. We refer to our MCTS approach with a system switch mechanism as MCTSwitch.

• **Instance-level switch.** This switch determines the thinking system for a given question based on the complexity evaluation of the question. We argue that simple questions can be directly handled with System 1 to ensure efficiency, while complex questions leverage System 2 to enhance quality.

• **Step-level switch.** If the question is determined to use System 2 thinking mode, we further employ the step-level switch to evaluate whether the next reasoning step requires System 1 or 2 to achieve a trade-off between the efficiency and effectiveness based on the complexity and uncertainty of the current context.

Switch Model Training. To achieve reliable system switch, we train a switch model by collecting a set of synthetic data. For the training data of step-level switch, we assign labels (0 or 1) to each node in the search tree based on its value. Specifically, we define a threshold γ: if a node's value exceeds γ, we label its state as 1, which denotes requiring System 2 thinking mode as a large value indicates a higher likelihood of generating hallucinated text. Conversely, we label a node with value below γ as 0, indicating that System 2 thinking mode is not required. Through this process, we obtain the thinking system labels for each step and utilize them as the training data for step-level switch. For instance-level data, we use the policy model to directly generate responses to the given question. The questions with correct responses are labeled as 0 (not requiring System 2 thinking mode), while those with incorrect responses are labeled as 1 (requiring System 2 thinking mode). These labeled questions serve as the training data for instance-level switch. The system switch model is then trained on the mixed instance-level and step-level training data using supervised fine-tuning, following the same training objective as the reward model.

Switch Model Inference. During inference, the system switch model first predicts the thinking system at the instance level. If the prediction is 0, the policy model adopts System 1 thinking mode to directly generate a response. If the prediction is 1, the policy model employs MCTS to perform deliberate reasoning. At the expansion step of MCTS, the switch model evaluates each node and predicts whether System 1 or 2 should be used. For nodes requiring System 1, a single sentence is generated as the child node; while for other nodes, the policy model follows the expansion process (Section 3.2) to generate multiple child nodes. The threshold γ controls the balance between System 1 and 2 thinking, optimizing efficiency and reasoning quality. The entire process is formalized in Algorithm 1 in Appendix A.

4 Experiments

4.1 Experimental Setup

Datasets and Metrics. We evaluate HaluSearch across multiple question-answering datasets in both English and Chinese. For English, we select HaluEval-QA (Li et al., 2023a), TruthfulQA (Lin et al., 2022), and SimpleQA (Wei et al., 2024). For Chinese, we select HalluQA (Cheng et al., 2023), ChineseSimpleQA (He et al., 2024), and Chinese-FactEval (Chern et al., 2023). We use *accuracy* as the evaluation metric. Specifically, we employ GPT-4 to assess the correctness of model-generated response by comparing it with the corresponding ground truth for each question.

Baselines. We select the following inference-stage hallucination mitigation methods as baselines for comparison. Additionally, we report the accuracy of direct generation by the policy models as the lower bounds for reference.

• **Chain-of-Thought** (Wei et al., 2022) prompts the model to generate intermediate reasoning steps before arriving at the final answer. In this work, we employ zero-shot CoT, which appends the phrase "Let's think step by step." into the prompt.

Methods	HaluEval-QA	TruthfulQA	SimpleQA	HalluQA	ChineseSimpleQA	ChineseFactEval
Llama3.1-8B-Instruct						
Direct Generation	35.60	24.50	3.00	8.25	15.50	29.60
CoT	36.80	33.50	4.00	6.80	19.50	32.00
SC	35.00	39.00	<u>6.00</u>	8.25	21.00	<u>36.00</u>
BoN	<u>37.80</u>	<u>43.50</u>	5.50	<u>12.62</u>	<u>29.00</u>	35.60
Self-Refine	28.20	30.50	5.00	8.25	17.50	33.60
ITI	35.20	37.50	2.00	-	-	-
MCTS	**45.40**	**47.50**	**8.50**	**16.50**	**30.50**	**40.80**
Qwen2-7B-Instruct						
Direct Generation	34.00	26.50	5.50	32.04	28.50	56.00
CoT	35.60	<u>38.00</u>	4.50	33.50	29.00	54.40
SC	37.20	36.00	4.00	32.52	30.50	59.20
BoN	<u>39.00</u>	37.50	<u>6.00</u>	<u>35.44</u>	<u>35.00</u>	<u>64.00</u>
Self-Refine	35.80	26.00	<u>6.00</u>	26.21	30.50	48.80
ITI	32.20	17.00	3.00	-	-	-
MCTS	**43.69**	**45.07**	**12.50**	**43.69**	**36.00**	**70.40**

Table 1: Evaluation results of **Llama3.1-8B-Instruct** and **Qwen2-7B-Instruct** on six English and Chinese datasets. **Bold** denotes the best results and <u>underline</u> denotes the second best results.

- **Self-Consistency** (Wang et al., 2023) samples multiple responses during inference and selects the most consistent response as the final answer.
- **Best-of-N** (Lightman et al., 2024) is similar to self-consistency, which selects the best response through a reward model.
- **Self-Refine** (Madaan et al., 2024) generates an initial response, evaluates it through feedback, and iteratively refines the response based on this feedback until a satisfactory version is achieved.
- **ITI** (Li et al., 2024b) operates by shifting model activations during inference to enhance the truthfulness of the generated responses.

Implementation Details. We evaluate our approach and the compared baselines using Llama3.1-8B-Instruct (Dubey et al., 2024) and Qwen2-7B-Instruct (Yang et al., 2024) as policy models, with GPT-4 serving as the reward model. In the MCTS process, we set the number of nodes expanded per step to 10, perform 5 rollouts for each node, and limit the maximum number of simulations to 20. In the UCT algorithm, the weight w is set to 0.4. For Self-Consistency and Best-of-N, we sample 20 responses per question. For ITI, we follow the original setting (Li et al., 2024b) by using models adjusted on the TruthfulQA dataset and report performance only on English datasets to ensure fairness. For all methods, the decoding temperature of the policy model is set to 0.9, with a 0-shot prompting configuration.

4.2 Main Results

The evaluation results of our method and the baselines are presented in Table 1.

Firstly, prompt-based generation methods demonstrate improved response accuracy compared to direct generation. However, the extent of this improvement is inconsistent and varies across tasks. For instance, on the TruthfulQA dataset, Chain-of-Thought prompting achieves an accuracy of 33.50% and Self-Refine attains 30.50% for Llama3.1-8B-Instruct, representing improvements of 9.00% and 6.00% compared to direct generation, respectively. However, on the HaluEval-QA dataset, these methods show less pronounced improvements, with CoT achieving 36.80% and Self-Refine performing even worse at 28.20%, which is affected by inherent capabilities and prompt sensitivity of the policy model.

Secondly, inference-time intervention methods exhibit limited generalization when adjusting activations on specific datasets. For ITI, the models we use are designed to probe and adjust activations on the TruthfulQA dataset, achieving effective improvements on this dataset (*e.g.,* 37.50% on the TruthfulQA dataset with Llama3.1-8B-Instruct as the policy model). However, this effectiveness diminishes on other datasets, often underperforming compared to direct generation. For instance, the accuracy of ITI drops to 2.00% on the SimpleQA dataset, demonstrating its limited transferability in scenarios with scarce data.

Reward Model	HaluEval-QA	TruthfulQA	SimpleQA
Llama3.1-8B-Instruct			
GPT-4 RM	45.40	47.50	8.50
Generative RM	42.60	45.50	7.50
Generative RM + Critic	46.20	50.00	7.85
Qwen2-7B-Instruct			
GPT-4 RM	43.69	45.07	12.50
Generative RM	40.40	40.00	6.50
Generative RM + Critic	42.80	50.50	8.50

Table 2: Evaluation results of different self-evaluation reward modeling approaches on Llama3.1-8B-Instruct and Qwen2-7B-Instruct.

Thirdly, strategies that generate multiple responses and leverage reward signals for selection can robustly enhance the quality of model outputs and effectively reduce hallucinations. Among all baselines, Self-Consistency and Best-of-N exhibit relatively strong performance across most datasets (*e.g.,* 36.00% accuracy on ChineseFactEval for SC and 39.00% on HaluEval-QA for BoN). In comparison, HaluSearch achieves the best performance across all six Chinese and English datasets. By leveraging MCTS to model the response generation process as step-by-step reasoning, our method provides fine-grained reward signals for each generation step, enabling effective guidance and a balance between exploration and exploitation. HaluSearch effectively reduces error accumulation and mitigates hallucinations in the final response, demonstrating the effectiveness of slow and deliberate reasoning during inference.

4.3 Reward Model Analysis

Beyond employing GPT-4 as the reward model, we conduct experiments to evaluate the performance of our trained self-evaluation reward models. Specifically, we investigate the two reward modeling approaches described in Section 3.3: (1) Generative RM, which directly generates numerical scores to responses during evaluation; (2) Generative RM + Critic, which first criticizes and analyzes the responses, and then generates a score based on the content of this feedback. We sample 1,000 examples from the HaluEval-QA dataset and 500 examples from the TruthfulQA dataset to generate training data, reserving the remaining data for evaluation. After filtering, we obtain 52K samples for reward data and 38K samples for critique data, which are used to train the reward model. Table 2 presents the results of employing Llama3.1-8B-Instruct or Qwen2-7B-Instruct as the policy model

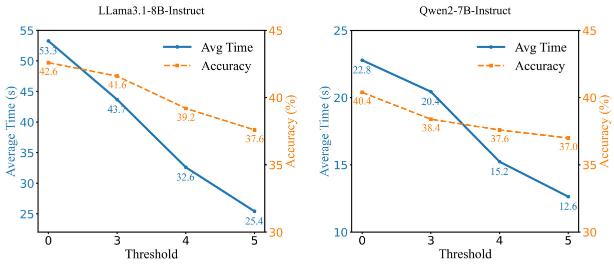

Figure 2: Impact of switch thresholds on time cost and accuracy on HaluEval-QA dataset.

and reward model (RM) compared to utilizing GPT-4 as reward model.

As we can see, Generative RM achieves competitive performance after training, outperforming other baselines shown in Table 1. Building on this, the Generative RM + Critic approach, which is trained using GPT-4-generated critiques and scores, shows substantial improvement over Generative RM. By introducing a critique step before scoring, it achieves significant gains, particularly on the TruthfulQA dataset (50.00% for Llama3.1-8B-Instruct and 50.50% for Qwen2-7B-Instruct), even surpassing the GPT-4 RM when it only provides scores. These results indicate that incorporating a critique step can improve the scoring accuracy of Generative RM, enabling more effective self-evaluation.

4.4 System Switch Analysis

To validate the effectiveness of the proposed system switch mechanism, we investigate the impact of slow thinking threshold γ (*i.e.,* the proportion of responses generated in slow thinking mode) on hallucination rates and response efficiency. Specifically, we collect 10K training data from HaluEval-QA and TruthfulQA, categorize them based on different score thresholds (*e.g.,* $\gamma = 3, 4, 5$), and label samples exceeding these thresholds for the slow thinking mode, as higher scores signify a higher likelihood of hallucination. We use the categorized datasets to train switch models optimized for different thresholds and evaluate their performance on the HaluEval-QA dataset. Here, both the reward model and the switch model are trained using Llama3.1-8B-Instruct and Qwen2-7B-Instruct, which also serve as the policy models.

In Figure 2, we present the accuracy of generated responses and the average inference time per question for different switch thresholds. When the switch threshold is set to 0, corresponding to a 100% slow thinking ratio, the model achieves its

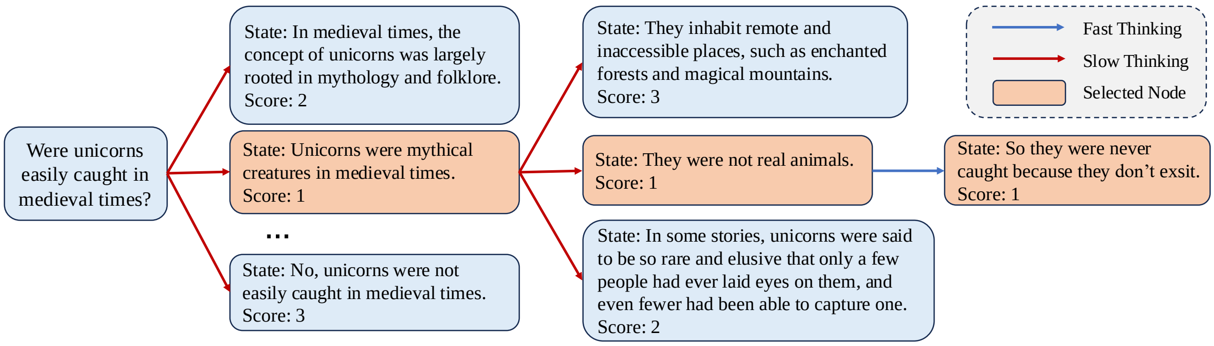

Figure 3: An example showing the deliberative reasoning process with system switch of HaluSearch in TruthfulQA.

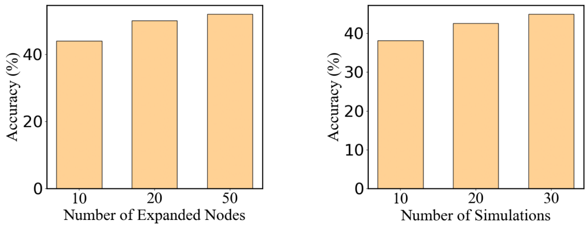

Figure 4: Results on HaluEval-QA *w.r.t.* the number of expansions (Left) or the number of simulations (Right).

highest accuracy of 42.6% with an average inference time of 53.3 seconds. As the switch threshold increases, the proportion of slow thinking gradually decreases, with only states exhibiting very high hallucination levels triggering slow thinking. This leads to a significant reduction in the average inference time per question, accompanied by a slight decrease in response accuracy. The trade-off between time efficiency and accuracy becomes more pronounced as the threshold increases. For instance, at a threshold of 5, where the slow thinking ratio is minimal, the average inference time per question drops to 25.4 seconds, while the accuracy remains at 37.6%. These observations demonstrate that the system switch mechanism effectively balances accuracy and efficiency, allowing for task-specific adjustments.

4.5 Further Analysis

Hyper-parameter Analysis. To validate the effectiveness of our proposed HaluSearch, we analyze the impact of its key hyper-parameters: the number of expanded nodes per step and simulations. Experiments are conducted on the HaluEval-QA dataset using Llama3.1-8B-Instruct as the policy model. We vary the number of expanded nodes in the set {10, 20, 50} while keeping the number of simulations fixed at 20, and vary the number of simulations in the set {10, 20, 30} while fixing the number of expanded nodes at 10. The results are shown in Figure 4. As we can see, increasing the number of expanded nodes and simulations improves the performance of HaluSearch. This improvement is attributed to the expanded search space, which increases the likelihood of identifying the correct answer by sampling more potential responses. However, as the number of expanded nodes and simulations increases further, the performance gains decrease due to the inherent limitations of the internal knowledge of the policy model and the scoring accuracy of the reward model.

Case Study. In Figure 3, we present an example from the TruthfulQA dataset to illustrate the reasoning process of HaluSearch. Starting with the question as the root node (*i.e., "Were unicorns easily caught in medieval times?"*), the policy model first selects slow thinking mode to generate multiple reasoning steps. While the model accurately identifies that unicorns are mythical creatures, it also generates erroneous reasoning paths, such as treating unicorns as real animals that could potentially be caught. The reward model evaluates these steps and assigns lower hallucination scores to accurate steps (*e.g., "Unicorns were mythical creatures in medieval times."*), guiding further expansion to reliable next steps (*e.g., "They were not real animals."*). With these intermediate steps, the model switches to fast thinking mode to efficiently derive the final answer: *"Unicorns don't exist."* This process illustrates how HaluSearch performs step-by-step reasoning and switches between thinking modes to generate more reliable responses.

5 Conclusion

In this work, we presented HaluSearch, a framework that integrates tree search-based algorithms (*e.g.,* MCTS) to enable explicit slow thinking process in LLMs for mitigating hallucinations.

HaluSearch frames text generation as a step-by-step reasoning process, guided by a self-evaluation reward model that scores each step and selects the most reliable generation path. For self-evaluation, we trained the reward model with two modeling approaches, using data synthesized by the HaluSearch framework. Moreover, to improve efficiency, we introduced a dynamic system switch mechanism, which utilizes a trained switch model to alternate between fast and slow thinking modes at both instance and step levels. Extensive experimental results demonstrated that our framework outperformed other inference-stage hallucination mitigation methods across a range of English and Chinese datasets.

Limitations

In HaluSearch, we employ MCTS as the tree search method within our framework. Future work can explore the integration of other tree search algorithms for further evaluation. Additionally, the system switch mechanism relies on the switch model, which may be influenced by the model's capabilities. In future work, we will investigate more advanced methods to determine the optimal thinking mode.

Acknowledgments

This work was partially supported by National Natural Science Foundation of China under Grant No. 92470205 and 62222215, Beijing Municipal Science and Technology Project under Grant No.Z231100010323009, Beijing Natural Science Foundation under Grant No. L233008. Xin Zhao is the corresponding author.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Sourav Banerjee, Ayushi Agarwal, and Saloni Singla. 2024. Llms will always hallucinate, and we need to live with this. *CoRR*, abs/2409.05746.

Qinyuan Cheng, Tianxiang Sun, Wenwei Zhang, Siyin Wang, Xiangyang Liu, Mozhi Zhang, Junliang He, Mianqiu Huang, Zhangyue Yin, Kai Chen, and Xipeng Qiu. 2023. Evaluating hallucinations in chinese large language models. *CoRR*, abs/2310.03368.

I-Chun Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, and Pengfei Liu. 2023. Factool: Factuality detection in generative AI - A tool augmented framework for multi-task and multi-domain scenarios. *CoRR*, abs/2307.13528.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2024. Chain-of-verification reduces hallucination in large language models. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 3563–3578. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 82 others. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Mohamed Elaraby, Mengyin Lu, Jacob Dunn, Xueying Zhang, Yu Wang, and Shizhu Liu. 2023. Halo: Estimation and reduction of hallucinations in open-source weak large language models. *CoRR*, abs/2308.11764.

Yancheng He, Shilong Li, Jiaheng Liu, Yingshui Tan, Weixun Wang, Hui Huang, Xingyuan Bu, Hangyu Guo, Chengwei Hu, Boren Zheng, Zhuoran Lin, Xuepeng Liu, Dekai Sun, Shirong Lin, Zhicheng Zheng, Xiaoyong Zhu, Wenbo Su, and Bo Zheng. 2024. Chinese simpleqa: A chinese factuality evaluation for large language models. *arXiv preprint arXiv:2411.07140*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *CoRR*, abs/2311.05232.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Jinhao Jiang, Zhipeng Chen, Yingqian Min, Jie Chen, Xiaoxue Cheng, Jiapeng Wang, Yiru Tang, Haoxiang Sun, Jia Deng, Wayne Xin Zhao, and 1 others. 2024. Technical report: Enhancing llm reasoning with reward-guided tree search. *arXiv preprint arXiv:2411.11694*.

Daniel Kahneman. 2011. Thinking, fast and slow. *Farrar, Straus and Giroux*.

Haoqiang Kang, Juntong Ni, and Huaxiu Yao. 2023. Ever: Mitigating hallucination in large language models through real-time verification and rectification. *CoRR*, abs/2311.09114.

Jikun Kang, Xin Zhe Li, Xi Chen, Amirreza Kazemi, and Boxing Chen. 2024. Mindstar: Enhancing math reasoning in pre-trained llms at inference time. *CoRR*, abs/2405.16265.

Levente Kocsis and Csaba Szepesvári. 2006. Bandit based monte-carlo planning. In *Machine Learning: ECML 2006, 17th European Conference on Machine Learning, Berlin, Germany, September 18-22, 2006, Proceedings*, volume 4212 of *Lecture Notes in Computer Science*, pages 282–293. Springer.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Junyi Li, Jie Chen, Ruiyang Ren, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2024a. The dawn after the dark: An empirical study on factuality hallucination in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10879–10899, Bangkok, Thailand. Association for Computational Linguistics.

Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023a. Halueval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6449–6464.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2024b. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023b. Textbooks are all you need II: phi-1.5 technical report. *CoRR*, abs/2309.05463.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Rensis Likert. 1932. A technique for the measurement of attitudes. *Archives of Psychology*.

Sheng-Chieh Lin, Luyu Gao, Barlas Oguz, Wenhan Xiong, Jimmy Lin, Wen-tau Yih, and Xilun Chen. 2024. FLAME: factuality-aware alignment for large language models. *CoRR*, abs/2405.01525.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3214–3252. Association for Computational Linguistics.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2024. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36.

Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, and 1 others. 2024. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *arXiv preprint arXiv:2412.09413*.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. 2024. Llms know more than they show: On the intrinsic representation of LLM hallucinations. *CoRR*, abs/2410.02707.

Vipula Rawte, Amit P. Sheth, and Amitava Das. 2023. A survey of hallucination in large foundation models. *CoRR*, abs/2309.05922.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *CoRR*, abs/2408.03314.

Xinyu Tang, Xiaolei Wang, Zhihao Lv, Yingqian Min, Wayne Xin Zhao, Binbin Hu, Ziqi Liu, and Zhiqiang Zhang. 2025. Unlocking general long chain-of-thought reasoning capabilities of large language models via representation engineering. *arXiv preprint arXiv:2503.11314*.

Xinyu Tang, Xiaolei Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2024. Dawn-icl: Strategic planning of problem-solving trajectories for zero-shot in-context learning. *arXiv preprint arXiv:2410.20215*.

Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher D. Manning, and Chelsea Finn. 2024. Fine-tuning language models for factuality. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Chaojie Wang, Yanchen Deng, Zhiyi Lv, Zeng Liang, Jujie He, Shuicheng Yan, and Bo An. 2024. Q*: Improving multi-step reasoning for llms with deliberative planning. *CoRR*, abs/2406.14283.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Peter C Wason and J St BT Evans. 1974. Dual processes in reasoning? *Cognition*, 3(2):141–154.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yiqing Xie, Wenxuan Zhou, Pradyot Prakash, Di Jin, Yuning Mao, Quintin Fettes, Arya Talebzadeh, Sinong Wang, Han Fang, Carolyn P. Rosé, Daniel Fried, and Hejia Zhang. 2024. Improving model factuality with fine-grained critique-based evaluator. *CoRR*, abs/2410.18359.

Ziwei Xu, Sanjay Jain, and Mohan S. Kankanhalli. 2024. Hallucination is inevitable: An innate limitation of large language models. *CoRR*, abs/2401.11817.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 43 others. 2024. Qwen2 technical report. *CoRR*, abs/2407.10671.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Hongbin Ye, Tong Liu, Aijia Zhang, Wei Hua, and Weiqiang Jia. 2023. Cognitive mirage: A review of hallucinations in large language models. *CoRR*, abs/2309.06794.

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. 2024. Distilling system 2 into system 1. *CoRR*, abs/2407.06023.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2023. Siren's song in the AI ocean: A survey on hallucination in large language models. *CoRR*, abs/2309.01219.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, and 3 others. 2023. A survey of large language models. *CoRR*, abs/2303.18223.

Appendix

A MCTSwitch Algorithm

We formalize and present our proposed dynamic system switch mechanism within MCTS (*i.e.,* MCTSwitch) in Algorithm 1.

Algorithm 1 MCTSwitch

Input: policy model π_θ, reward model R, switch model σ_s, number of expansions K, number of rollouts m, UCT weight w, max iterations M, reward threshold r_{th}, query q
Initialize: Root $s_0 \leftarrow q$, $\mathcal{C}(s_0) = \emptyset$, $t \leftarrow 0$
Instance Switch: instance_mode $\leftarrow \sigma_s(q)$
if instance_mode is slow **then**
    while $t < M$ **do**
        $s_t \leftarrow \text{Select}(\text{UCT}(s_0, w))$
        Step Switch: step_mode $\leftarrow \sigma_s(s_t)$
        if step_mode is slow **then**
            $\mathcal{C}(s_t) \leftarrow \text{Expand}(s_t, \pi_\theta, K)$
            for $s_c \in \mathcal{C}(s_t)$ **do**
                $\mathcal{C}_{\text{r}}(s_c) \leftarrow \text{Rollout}(s_c, \pi_\theta, m)$
                $r_{s_c} \leftarrow \text{Avg}(\mathcal{C}_{\text{r}}(s_c), R)$
                if $r_{s_c} \geq r_{\text{th}}$ and s_c is terminal **then**
                    Break
                end if
            end for
        else
            $s_c \leftarrow \pi_\theta(s_t)$
            $\mathcal{C}_{\text{r}}(s_c) \leftarrow \text{Rollout}(s_c, \pi_\theta, m)$
            $r_{s_c} \leftarrow R(s_c)$
            if $r_{s_c} \geq r_{\text{th}}$ and s_c is terminal **then**
                Break
            end if
        end if
        $\text{Backpropagate}(s_0, \mathcal{C}(r_{s_c}))$
        $t \leftarrow t + 1$
    end while
    $A \leftarrow \text{BestNode}(s_0)$
else
    $A \leftarrow \pi_\theta(q)$
end if
Output: A

B Prompts

The detailed scoring instructions for the reward model are presented in Table 3. During the reward data generation phase, the correct answers (highlighted in gray) are provided, whereas they are excluded during the evaluation phase.

Please rate the likelihood of hallucinations (wrong, irrelevant, unfounded, or contradictory content) appearing in the continuation of the current answer fragment. There are five levels of hallucination probability:
1 - No hallucination risk: Future content will be entirely accurate, relevant, and well-supported.
2 - Low hallucination risk: Future content is likely to be accurate and relevant, with minor uncertainties possible.
3 - Moderate hallucination risk: Some hallucinations, such as minor inaccuracies or unclear relevance, may appear, but the content will still be mostly reliable.
4 - High hallucination risk: Future content will likely contain noticeable hallucinations, such as errors, irrelevant information, or contradictions, reducing reliability.
5 - Very high hallucination risk: Future content is highly likely to include significant hallucinations, such as major errors, contradictions, or fabricated information, making it highly unreliable.
Please output a score from 1 to 5. The higher the score, the higher the probability of hallucinations. Only output the score without any further explanation. (Output the score after your analyses.) Do not judge a reply as hallucinated just because it is incomplete.
We provide the correct answer as a reference.
Question:
Correct Answer: *(Only provided when generating reward data.)*
Generated Answer:
Score:

Table 3: Detailed scoring instructions for the reward model. The gray parts are provided only during the generation of reward data and are excluded during evaluation.