Supervised Optimism Correction: Be Confident When LLMs Are Sure

Junjie Zhang[1*], **Rushuai Yang**[2*], **Shunyu Liu**[1], **Ting-En Lin**[3], **Fei Huang**[3],
Yi Chen[2], **Yongbin Li**[3†], **Dacheng Tao**[1†]

[1]Nanyang Technological University, Singapore
[2]Hong Kong University of Science and Technology
[3]Tongyi Lab

Abstract

In this work, we establish a novel theoretical connection between supervised fine-tuning and offline reinforcement learning under the token-level Markov decision process, revealing that large language models indeed learn an implicit Q-function for inference. Through this theoretical lens, we demonstrate that the widely used beam search method suffers from unacceptable over-optimism, where inference errors are inevitably amplified due to inflated Q-value estimations of suboptimal steps. To address this limitation, we propose ***S**upervised **O**ptimism **C**orrection* (SOC), which introduces a simple yet effective auxiliary loss for token-level Q-value estimations during supervised fine-tuning. Specifically, the auxiliary loss employs implicit value regularization to boost model confidence in expert-demonstrated responses, thereby suppressing over-optimism toward insufficiently supervised responses. Extensive experiments on mathematical reasoning benchmarks, including GSM8K, MATH, and GAOKAO, showcase the superiority of the proposed SOC with beam search across a series of open-source models.

1 Introduction

Recent advances in Large Language Models (LLMs) have demonstrated remarkable success across diverse tasks such as instruction following (Brown et al., 2020; Zhou et al., 2023; Taori et al., 2023; Tao et al., 2024; Liu et al., 2025b), code generation (Liu et al., 2023; Le et al., 2022; Nijkamp et al., 2022; Jiang et al., 2024), and medical diagnosis (Zhang et al., 2023b; Wang et al., 2023). Within these developments, complex reasoning capabilities have attracted increasing attention from research communities, attributed to their capability of enabling LLMs to tackle intricate problem-solving (Wei et al., 2022; Yao et al., 2022; Kojima et al., 2022). Despite these achievements, learning to reason remains a critical yet challenging task for LLMs, particularly for smaller models with limited parameters (Liu et al., 2025a; Zhang et al., 2025b,a). The inherent complexity stems from inefficient exploration of LLMs, as the combinatorial nature of the vocabulary space results in an exponential growth of potential reasoning paths (Snell et al., 2024).

A prevalent paradigm for enhancing reasoning capabilities involves Supervised Fine-Tuning (SFT) on high-quality demonstration data (Brown et al., 2020; Yao et al., 2024; Yang et al., 2024a), where models learn to imitate expert reasoning patterns through next-token prediction. The complementary strategies employ search-based decoding techniques during inference (Snell et al., 2024; Xie et al., 2024b; Ding et al., 2025), such as beam search, which aims to enhance reasoning by exploring multiple candidate pathways. However, there is often an overlooked disconnect between the local token-level optimization of SFT and the global sequence-level objectives pursued by search-based decoding. This disconnect results in a critical misalignment: while SFT focuses on maximizing the likelihood of individual token predictions, search-based decoding operates by scoring entire sequences, which may not directly align with the local training objectives of LLMs. By addressing this gap, we can potentially unlock more robust reasoning capabilities in LLMs, aligning training objectives more closely with inference-time goals.

In this work, we employ the token-level Markov decision process to establish a novel theoretical connection between SFT and offline Reinforcement Learning (RL) (Levine et al., 2020; Kumar et al., 2020; Kostrikov et al., 2021; Lyu et al., 2022). We theoretically demonstrate that LLMs indeed learn an implicit Q-function to estimate the expected utility of token sequences during SFT. Through this lens, we further explore the over-optimism problem in the widely used beam search method, re-

[*]Equal contribution.
[†]Corresponding authors.

Findings of the Association for Computational Linguistics: ACL 2025, pages 8867–8880
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

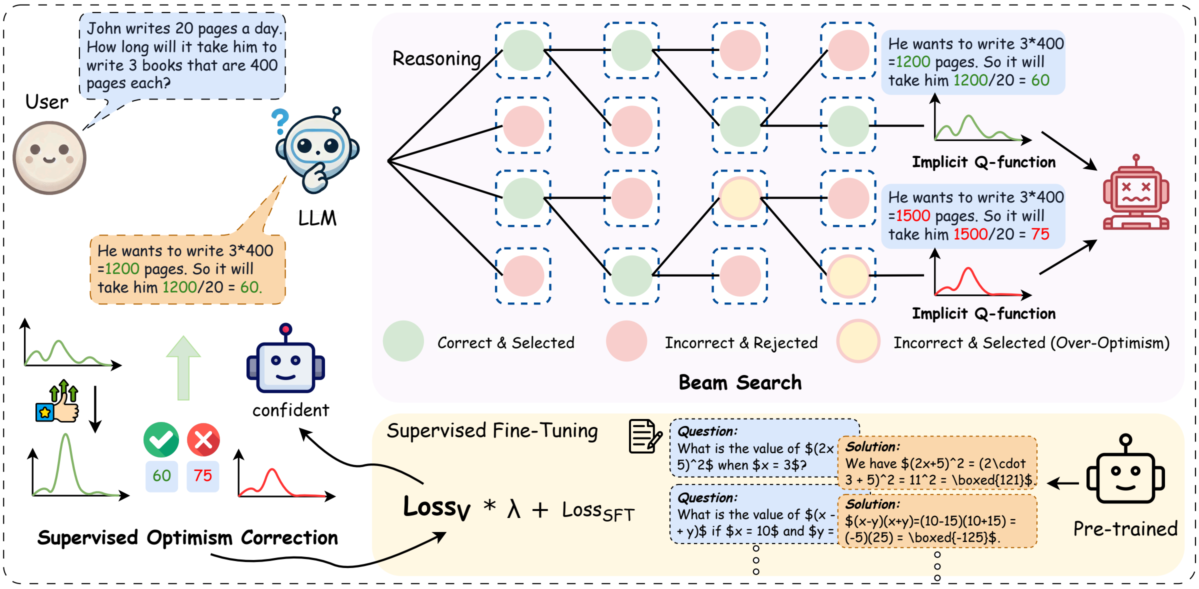

John writes 20 pages a day. How long will it take him to write 3 books that are 400 pages each?
User
LLM
He wants to write 3*400 =1200 pages. So it will take him 1200/20 = 60.
confident
60
75
Supervised Optimism Correction
Reasoning
He wants to write 3*400 =1200 pages. So it will take him 1200/20 = 60
Implicit Q-function
He wants to write 3*400 =1500 pages. So it will take him 1500/20 = 75
Implicit Q-function
Correct & Selected
Incorrect & Rejected
Incorrect & Selected (Over-Optimism)
Beam Search
Supervised Fine-Tuning
Loss$_V$ * λ + Loss$_{SFT}$
Question:
What is the value of $(2x + 5)^2$ when $x = 3$?
Solution:
We have $(2x+5)^2 = (2\cdot 3 + 5)^2 = 11^2 = \boxed{121}$.
Question:
What is the value of $(x - y)(x + y)$ if $x = 10$ and $y =
Solution:
$(x-y)(x+y)=(10-15)(10+15) = (-5)(25) = \boxed{-125}$.
Pre-trained

Figure 1: Illustration of Supervised Optimism Correction. Guided by an implicit Q-function, beam search suffers from the over-optimism problem during the decoding process, which confuses LLM for reliable response choice. In particular, the over-optimism can amplify errors during beam search, leading to the selection of incorrect trajectories with higher Q-values. To alleviate this problem, SOC introduces an auxiliary loss during SFT to boost model confidence in expert-demonstrated responses, about which LLM should be sure.

vealing that the search process disproportionately favors sequences with inflated Q-value estimations. This over-optimism arises because the beam search method autoregressively selects tokens with locally overestimated Q-values, thereby inevitably amplifying errors through cascading suboptimal steps.

To alleviate this problem, we propose Supervised Optimism Correction (SOC), which introduces a simple auxiliary loss during SFT to give supervised responses a state-value bonus. Unlike prior RL-based methods that require explicit reward modeling, SOC operates purely within the SFT paradigm by imposing implicit value regularization. This regularization boosts confidence in expert-demonstrated data while potentially penalizing high Q-value estimations for insufficiently supervised reasoning steps. As a result, the model learns to autonomously prune low-quality reasoning paths during inference without relying on external verifiers or reward models. For instance, when encountering erroneous intermediate steps, the corrected Q-values suppress further exploration of those branches, mirroring human-like error recognition and recovery patterns, requiring no architectural modifications or additional inference-time computations. Our core contributions are summarized as follows:

- We establish a novel theoretical connection between SFT and offline RL, identifying and formalizing the over-optimism problem of implicit Q-functions for LLM beam search.
- We develop SOC, a lightweight yet effective method that boosts model confidence in expert-demonstrated data through an auxiliary value regularization loss during SFT.
- Extensive experiments demonstrate the effectiveness of SOC in mathematical reasoning benchmarks GSM8K, MATH, and GAOKAO, significantly improving the performance of open-source models like Qwen-2-1.5B, Qwen-2.5-3B, and Qwen-2.5-7B.

2 Preliminaries

2.1 Token-level MDP for LLMs

We start by formulating the token generation process of LLMs as a token-level Markov Decision Process (MDP) (Rafailov et al., 2024b,a; Zhong et al., 2024, 2022), enabling a structured analysis of their decision-making dynamics and reasoning capabilities. An MDP (Sutton and Barto, 1998) is typically defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, $\mathcal{P}$ the transition probability function, R the reward function, and $\gamma \in [0, 1]$ the discount factor, adopting a discount factor of $\gamma = 1$ throughout the paper. In our formulation, the state $s \in \mathcal{S}$ corresponds to the token generation context at step t, represented as $s_t = (x_0, x_1, \dots, x_t)$, which serves as the conditioning context for generating the next token x_{t+1}. The action space $\mathcal{A}$ is defined by the fixed vocabulary from which the next token is selected. The probability distribution over actions is parameterized by the LLM's learned policy $\pi_\theta(a_t|s_t)$, with $a_t \in \mathcal{A}$ at each step t. In addition, the transition function $P(s_{t+1}|s_t, a_t)$ models the concatenation of the current state with the chosen action to form the subsequent state, *i.e.* $s_{t+1} = (s_t, a_t)$. In the offline RL setting (Levine et al., 2020; Kumar et al., 2020; Kostrikov et al., 2021), the objective is to obtain a parameterized policy π_θ that maximizes the expected cumulative reward using an offline dataset $\mathcal{D}$. This can be formulated as:

$$\max_\theta \mathbb{E}_{s_t \sim \mathcal{D}, a_t \sim \pi_\theta} \left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t) \right], \quad (1)$$

where the reward function $R(s_t, a_t)$ measures the quality of the generated output. In tasks such as mathematical reasoning, the reward is often sparse, with a terminal reward $R_{\text{outcome}} = 1$ indicating the correctness of the final result, while intermediate rewards are set to zero. When adopting a discount factor of $\gamma = 1$, the gradient of Equation (1) with respect to the policy parameters is given by:

$$\mathbb{E}_{s_t \sim \mathcal{D}, a_t \sim \pi_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) R(\tau) \right]. \quad (2)$$

According to the sparse reward assumption in the above MDP setting for correct τ:

$$R(\tau) := \sum_{t=0}^{T} \gamma^t R(s_t, a_t) = R_{\text{outcome}} = 1, \quad (3)$$

then Equation (2) can be simplified to:

$$\mathbb{E}_{s_t \sim \mathcal{D}, a_t \sim \pi_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \right]. \quad (4)$$

On the other hand, the pretrained model π_θ in the SFT stage aims to imitate the behavior of the expert policy $\pi^\star$. This can be expressed as

$$\pi_\theta = \operatorname*{argmax}_\theta \mathbb{E}_{(s^\star, a^\star) \sim \mathcal{D}} \left[\log \pi_\theta(a^\star|s^\star) \right]. \quad (5)$$

Note that the optimization problem above is a special case of the optimization problem encountered in offline reinforcement learning. The reward signal comes from matching the expert's demonstrations. In addition, to avoid repetitive or suboptimal outputs, we can maximize the entropy of the policy $\mathcal{H}(\pi_\theta)$ simultaneously to encourage exploration and prevent the policy from becoming overly deterministic, the optimization problem will be written as

$$\pi_\theta = \operatorname*{argmax}_\theta \mathbb{E}_{(s^\star, a^\star) \sim \mathcal{D}} \left[\log \pi_\theta(a^\star|s^\star) \right] + \mathcal{H}\left(\pi_\theta \left(\cdot|s^\star \right) \right), \quad (6)$$

which is equivalent to a maximum entropy RL objective (Ziebart, 2010; Haarnoja et al., 2018) but with the reward coming from the expert's actions rather than an environment-based reward signal.

2.2 Deriving the Q-function as LLM Logits

In the general maximum entropy RL setting, the fixed-point solution of Equation (6) is given by (Ziebart, 2010; Rafailov et al., 2024a; Guo et al., 2021) as:

$$\pi^\star(a|s) = \exp\left(Q^\star(s, a) - V^\star(s) \right), \quad (7)$$

where $Q^\star(s, a)$ is the optimal Q-function, representing the accumulated reward starting from state s, taking action a, and following the optimal policy thereafter. The optimal value function $V^\star(s)$ is related to $Q^\star(s, a)$ and is given by:

$$V^\star(s) = \log \sum_a \exp(Q^\star(s, a)). \quad (8)$$

By combining Equation (7) and Equation (8) and taking the logarithm of both sides, we obtain:

$$\pi^\star(a|s) = \frac{\exp(Q^\star(s, a))}{\sum_{a'} \exp(Q^\star(s, a'))}. \quad (9)$$

Thus, the optimal policy is derived from the softmax of the corresponding Q-function. On the other hand, in the context of pretrained models in LLMs, the policy is also obtained from the softmax of logits:

$$\pi_\theta(a|s) = \frac{\exp(Q_\theta(s, a))}{\sum_{a'} \exp(Q_\theta(s, a'))}, \quad (10)$$

where $Q_\theta(s, a)$ represents the logits generated by the pretrained model. These logits implicitly approximate the optimal Q-function, ensuring that the parameterized policy π_θ closely resembles the optimal policy $\pi^\star$. We will refer to LLM logits as the implicit Q-function throughout the paper.

2.3 Beam Search Decoding

Beam search is a heuristic search algorithm widely used in decoding for LLMs' test time (Pascual et al., 2020; Sun et al., 2023; Liu et al., 2025a). Given a pre-trained language model π_θ that generates tokens in an autoregressive manner, standard beam search aims to approximate the most probable output sequence by maintaining a fixed-size set of candidate sequences (beams) at each decoding step. Formally, at each step t, the method expands all hypotheses by considering the top-k best candidate tokens according to their accumulated log-probability scores. This process continues until an end-of-sequence (EOS) token is generated or the maximum sequence length T is reached. The algorithm is presented in Algorithm 1.

Algorithm 1 Standard Beam Search Decoding

```
Require: Language model $\pi_\theta(a_t|s_t)$, question input $x_0$, beam width $k$, maximum sequence length $T$
Initialize beam set $\mathcal{B} \leftarrow \{(s_0, v_0)\}$, where $s_0 = x_0$ and score $v_0 = 0$
for $t = 1$ to $T$ do
    for each $(s_t, v_t) \in \mathcal{B}$ do
        Compute next-token probabilities $\pi_\theta(a_t \mid s_t)$ for $a_t \in \mathcal{A}$
        Select the top-$k$ tokens $\mathcal{A}_t^{\text{top}}$ from $\mathcal{A}$ based on the accumulated score:
            $v_{t+1} = v_t + \log \pi_\theta(a_t|s_t)$
        for each $a_t \in \mathcal{A}_t^{\text{top}}$ do
            Compute new state $s_{t+1} = (s_t, a_t)$
            Add corresponding $(s_{t+1}, v_{t+1})$ to $\mathcal{B}$
        end for
    end for
end for
Return best sequence: $\text{argmax}_{(s_T, v_T) \in \mathcal{B}} v_T$
```

3 Supervised Optimism Correction

In Subsection 3.1, we first discuss the over-optimism problem observed in beam search during inference. Next, Subsection 3.2 investigates a potential cause of this issue by analyzing how inflated Q-value estimation errors, particularly those arising from insufficiently supervised states, can amplify over-optimism through the maximization operation of beam search. Finally, Subsection 3.3 introduces our proposed Supervised Optimism Correction (SOC) method, which incorporates an auxiliary V loss to better align the response selection process with expert-demonstrated responses.

3.1 Over-optimism Problem in Beam Search

In this section, we explore the over-optimism problem in beam search, examining the factors that contribute to its exacerbation and its impact on response quality during inference. Beam search is widely used to generate sequences based on accumulated log-probability scores. However, over-optimism can arise when the search disproportionately favors sequences with inflated Q-value estimates, leading to suboptimal results. This phenomenon is also prevalent in traditional reinforcement learning contexts (Thrun and Schwartz, 2014; Van Hasselt et al., 2016; Kumar et al., 2020; Kostrikov et al., 2021; Wen et al., 2024b). To formalize this issue, we express the beam search selection process in terms of the Q-function. At every intermediate step T, by incorporating Equation (9), the accumulated log-probability over the sequence can be expressed as:

$$\sum_{t=0}^{T} \log \pi(a_t|s_t) = \sum_{t=0}^{T} (Q(s_t, a_t) - V(s_t)) \tag{11}$$

$$= \sum_{t=0}^{T-1} (Q(s_t, a_t) - V(s_{t+1})) + Q(s_T, a_T) - V(s_0). \tag{12}$$

This can be further simplified using the Bellman equation for the value function for all $t < T$:

$$Q(s_t, a_t) = \mathbb{E}_{s_{t+1}}[R(s_t, a_t) + \gamma V(s_{t+1})]. \tag{13}$$

Under the assumptions that the transition dynamics $P(s_{t+1}|s_t, a_t)$ are deterministic, $\gamma = 1$, and intermediate rewards are zero by default, the accumulated log-probability simplifies to:

$$\sum_{t=0}^{T} \log \pi(a_t|s_t) = Q(s_T, a_T) - V(s_0). \tag{14}$$

Since all candidates share the same question input s_0, $V(s_0)$ is same and beam search relies heavily on these Q-values of the generated token to select the top-k candidates at each step, as described in Equation 11. However, the process may include candidates with inaccurate Q-value estimates $Q(s_T, a_T)$, which can result from limited supervision during the fine-tuning stage. When Q-values are overestimated in insufficiently supervised states, this overestimation can amplify errors during beam search, leading to the selection of suboptimal trajectories with higher Q-values, even if they are not aligned with the most reliable, well-supervised paths. Consequently, the final output is biased toward these less reliable paths. Figure 1 illustrates a case example of the over-optimism problem. In the next subsection, we investigate the causes of over-optimism in beam search, with a particular focus on how Q-value estimation errors contribute to its exacerbation and the resulting impact on beam search performance.

3.2 Impact of Value Function Estimation Error on Beam Search Performance

To analyze how the estimation error of Q affect the sampling process in inference time, we start to obtain the key observation from the gradient of the cross-entropy loss. Recall that in SFT stage, the model is trained to align its predicted distribution π_θ with the target distribution $\pi^\star$ by minimizing the cross-entropy loss (Le et al., 2022; Yang et al., 2024b):

$$\mathcal{L}_{\text{SFT}} = \mathbb{E}_{s\sim\mathcal{D}}\left[-\sum_a \pi^\star(a|s)\log\pi_\theta(a|s)\right], \tag{15}$$

where target distribution $\pi^\star$ could be one-hot encoding as dataset $\mathcal{D}$ indicates the set of high-quality demonstrations. By incorporating Equation (9) into the gradient form of $\mathcal{L}_{\text{SFT}}$, we can see that the $\mathcal{L}_{\text{SFT}}$ minimize the estimation error of implicit Q, *i.e.*

$$\begin{aligned}\nabla_\theta\mathcal{L}_{\text{SFT}} &= \mathbb{E}_{s\sim\mathcal{D}}\left[-\sum_a \frac{\pi^\star(a|s)}{\pi_\theta(a|s)}\frac{\partial\pi_\theta(a|s)}{\partial\theta}\right]\\ &=\mathbb{E}_{s\sim\mathcal{D}}\left[\sum_a \underbrace{\left(\widetilde{Q}_\theta(s,a)-\widetilde{Q}^\star(s,a)\right)}_{\text{estimation error}}\frac{\partial Q_\theta(s,a)}{\partial\theta}\right],\end{aligned} \tag{16}$$

where $\widetilde{Q}(s,a)\in[0,1]$ refers to normalized value function for each action a. We give the detailed proof in Appendix B. Intuitively, the learned policy π_θ is derived from the softmax of $Q_\theta(s,a)$, so estimation errors in $Q_\theta(s,a)$ will lead to deviations from the optimal policy $\pi^\star$, and the policy may favor suboptimal actions, leading to insufficiently supervised action and misguide to the suboptimal state. This error could inevitably be amplified during inference since the beam search takes action based on the estimated $Q_\theta(s,a)$ as shown in Equation (11).

3.3 V Loss for Optimism Correction

Based on the analysis in the previous subsection, one approach to mitigating the over-optimism problem is to reduce the impact of estimation error. To this end, we propose an auxiliary objective V loss, defined as:

$$\mathcal{L}_V = \mathbb{E}_{s\sim\mathcal{D}}\left[-\log\sum_a \exp Q_\theta(s,a)\right]. \tag{17}$$

Intuitively, this auxiliary loss serves to boost the overall implicit value function for high-quality, labeled data. By elevating the value estimates associated with supervised trajectories, the model is more likely to preferentially select these trajectories during inference, thereby counteracting the bias introduced by overestimated Q values in insufficiently supervised states. Overall, our total objective in the training stage can written as:

$$\mathcal{L}_{\text{overall}} = \mathcal{L}_{\text{SFT}} + \lambda\cdot\mathcal{L}_V \tag{18}$$

with a tuning hyperparameter λ.

3.4 Effect of the V Loss Update

To gain a mechanistic understanding of the V loss, we analyze the gradient of its objective function:

$$\nabla_\theta\mathcal{L}_{\text{V}} = \mathbb{E}_{s\sim\mathcal{D}}\left[-\sum_a \widetilde{Q}_\theta(s,a)\frac{\partial Q_\theta(s,a)}{\partial\theta}\right]. \tag{19}$$

This expression reveals how the update is guided by the weighted sum of the Q-function gradients for labeled data, where $\widetilde{Q}_\theta(s,a)$ acts as an implicit weighting factor influencing parameter adjustments. We provide an illustrative example for the update effect between $\mathcal{L}_{\text{SFT}}$ and $\mathcal{L}_{\text{V}}$ in Figure 2. Specifically, since $\sum_a \widetilde{Q}_\theta(s,a) = 1$ for all state s, A larger $\widetilde{Q}_\theta(s,a)$ for a specific action increases the magnitude of its corresponding gradient contribution, thereby resulting in a more substantial update for $Q_\theta(s,a)$. Consequently, actions with higher $\widetilde{Q}_\theta(s,a)$ experience a more significant increase in their Q values during optimization. Compared to the SFT update for a supervised action, the update magnitude induced by the V loss is different. In particular, when $Q(s,a)$ is close to $Q^\star(s,a)$, the gradient from the SFT objective tends to be small, as shown in Equation (16), potentially resulting in only minimal updates. In contrast, the V loss continues to increase $Q(s,a)$ for the supervised action, thereby compensating for the estimation gap along the supervised trajectories with appropriated λ.

4 Theoretical Analysis

In this section, we provide a theoretical analysis of our objective.

Theorem 4.1 (Contraction of Value Differences)**.** *Let $V_\theta(s)$ be the approximate value function of state s. Suppose that the value function is always positive for any state s. If the objective in SFT includes a additional term in Equation (17), then after one step of gradient descent with learning rate $\alpha \in [0,1]$, the gap between adjacent states' values contracts, i.e.,*

$$|V'_\theta(s_t) - V'_\theta(s_{t+1})| \leq |V_\theta(s_t) - V_\theta(s_{t+1})|, \quad \forall t,$$

where V' denotes the updated value function after one optimization step.

We provide a detailed proof in Appendix A. Theorem 4.1 states that the auxiliary loss encourages the implicit value function of neighboring states to become closer. To further illustrate the benefit of this regularization, consider the policy evaluation problem in a token-level MDP with sparse rewards, where the agent only receives a reward at the end of the trajectory while all intermediate rewards are zero. In this setting, the optimal value function satisfies the Bellman equation:

$$V^\star(s_t) = r\left(s_t, \pi^\star(a_t|s_t)\right) + V^\star(s_{t+1}). \tag{20}$$

Since the reward is zero for all intermediate states, it follows that:

$$V^\star(s_t) = V^\star(s_{t+1}), \forall t \text{ (intermediate states).} \tag{21}$$

Thus, in sparse reward settings, the optimal gap between the values of adjacent states is naturally small. By minimizing this gap during training, the auxiliary loss guides the learned value function to better reflect the underlying structure of the sparse reward MDP, rather than artificially transforming the sparse reward into a dense one, thereby facilitating more stable and efficient policy evaluation for sparse reward. Moreover, in sparse reward settings, value information needs to propagate over many steps. By encouraging smoother value estimates between adjacent states, the auxiliary loss effectively reduces variance in value updates, which can lead to more robust value estimation (Schulman et al., 2015, 2017).

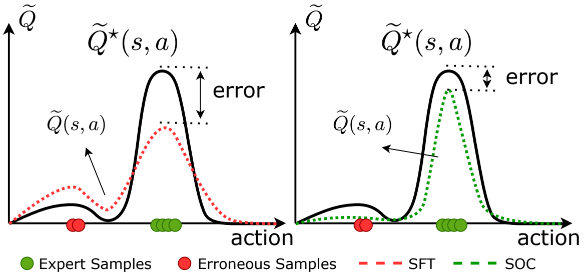


Figure 2: An illustrative example of gradient update magnitude on Q-value estimation in SFT and SOC. (Left) In SFT, small gradient updates lead to insufficient supervision, causing persistent overestimation errors and favoring suboptimal actions (red spots). (Right) SOC mitigates overestimation by aligning estimated values with expert demonstrations (green spots), reducing errors, and improving decision-making in inference times.

5 Experiments

In this section, we empirically demonstrate the effectiveness of SOC on the mathematical reasoning ability of LLMs. In Subsection 5.1 and 5.2, we introduce the datasets and setup of our experiments. We present the results of LLMs scaling from 1.5B to 7B parameters in Subsection 5.3 and provide an ablation study in Subsection 5.4.

5.1 Datasets

We conduct experiments on mathematics problem datasets: (1) GSM8K (Cobbe et al., 2021), which consists of 8.8K high quality linguistically diverse grade school math word problems split into 7.5K training set and 1.3K test set, and (2) MATH (Hendrycks et al.), which consists of problems from mathematics competitions with 5 difficulty level with 7.5K training set and 5.0K test set. (3) We additionally include a recently released dataset STEP-DPO-10K (Lai et al., 2024) as the training dataset.

In the experiments, we adopt a filtered STEP-DPO-10K and a filtered MATH dataset as the training set. Although the STEP-DPO-10K is curated to demonstrate a step-level preference of solutions, we only chose the full and preferred samples and regard them as supervised demonstrations as they are correct and complete question and answer pairs, consisting of 10.3K samples. Correspondingly, we chose the samples with the highest difficulty level (level 5) as the training dataset of MATH, which consists of 2.8K samples. Hereafter, we refer to the filtered STEP-DPO-10K and the filtered MATH as DATA-α and DATA-β. As many previous works, we use the GSM8K, MATH-500 (Lightman et al., 2023), and GaoKao2023En (Liao et al., 2024) test sets for evaluation.

Table 1: Results (accuracy % of pass@1) on mathematical reasoning tasks. We report the results of each model sampling with beam search (beam width of 5) and greedy decoding. For results of GSM8K and MATH-500, models are prompted with 4-shot demonstrations; for GAOKAO-EN, it is zero-shot. There is no sampling process in both beam search and greedy decoding, and therefore, the results are deterministic.

MODEL	GSM8K		MATH-500		GAOKAO-EN	
	BEAM	GREEDY	BEAM	GREEDY	BEAM	GREEDY
Qwen-2-1.5B-Base	56.1	48.7	25.6	23.6	19.5	15.8
+SFT(DATA-β)	54.9	56.4	24.0	20.2	17.7	16.1
+SFT(DATA-β)+SOC	56.0(+1.1)	55.6	25.4(+1.4)	21.6	18.2(+0.5)	17.1
+SFT(DATA-α)	69.1	64.6	25.0	24.0	23.9	22.3
+SFT(DATA-α)+SOC	69.4(+0.3)	64.3	26.6(+1.6)	24.4	27.3(+3.4)	23.4
Qwen-2.5-3B-Base	51.5	77.0	40.8	45.0	31.4	41.8
+SFT(DATA-β)	52.5	76.3	40.0	43.4	30.1	41.3
+SFT(DATA-β)+SOC	75.4(+22.9)	69.4	41.6(+1.6)	36.0	30.4(+0.3)	31.7
+SFT(DATA-α)	79.2	81.7	49.6	49.6	34.8	46.5
+SFT(DATA-α)+SOC	83.5(+4.3)	80.8	54.4(+4.8)	48.4	46.5(+11.7)	43.9
Qwen-2.5-7B-Base	58.7	85.3	8.8	54.6	33.0	47.3
+SFT(DATA-β)	79.9	84.0	41.0	47.2	44.9	44.2
+SFT(DATA-β)+SOC	83.7(+3.8)	83.5	41.4(+0.4)	48.4	45.7(+0.8)	44.2
+SFT(DATA-α)	54.7	87.3	41.4	56.0	28.6	51.9
+SFT(DATA-α)+SOC	87.9(+33.2)	86.7	60.4(+19.0)	53.4	51.4(+22.8)	49.1

5.2 Experimental Setup

In our experiments, we train a series of Qwen base models, Qwen-2-1.5B-Base, Qwen-2.5-3B-Base, and Qwen-2.5-7B-Base. We conduct SFT with high-quality math demonstrations to activate the base models' math reasoning ability to handle math problems. The DATA-α and DATA-β datasets are used independently as two distinct training settings to show the robustness of our method.

We utilize the accuracy on the test set of GSM8K, MATH-500, and GaoKao2023En as the evaluation metric. Specifically, LLMs are prompted with a few shots CoT of math problem solutions and output format requirements, such as generating the final answer in the boxed{}. LLMs predict the solutions with beam search and we compare the predicted answer with the ground truth answer to calculate the accuracy. Following the previous work (Yang et al., 2024b), we use the same few shots CoT prompt and set the maximum generation length as 2048. Also, we apply the chat template for structural format during the training, which consists of some special tokens like <|im_start|> and <|im_end|>. For evaluation, we use two kinds of CoT prompts (refer to Appendix C for details), one consists of the special tokens of SFT, and another does not. We report the better results of SFT models among the two prompts and use the same prompt for SOC, which ensures a fair comparison and reliable results regarding the impact of prompts.

5.3 Main Results

The goal of our empirical results is to show the effectiveness of SOC by improving the performance of beam search. In the main experiments, we SFT the base models by DATA-α and DATA-β with auxiliary loss of SOC. The results are shown in Table 1. SOC significantly improves the beam search performance over SFT on both training datasets, evaluated on three benchmarks. Despite its simplicity, SOC consistently improves the performance of SFT models in beam search decoding and gains even more than 20 percent accuracy improvement in several settings by generating more confident and reliable reasoning steps during the beam search.

From the results of the base models, we find that beam search has far worse performance than greedy decoding (refer to the example in Subsection 5.4). This issue stems from inaccurate Q-value estimation. It exerts an insignificant impact on greedy sampling if it does not alter the action of each individual step. However, it significantly degrades beam search results because it influences evaluation of the entire sequence and gets amplified during

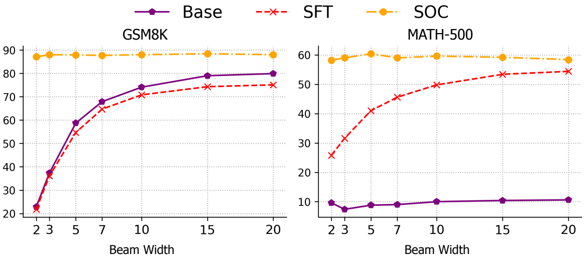


Figure 3: Performance of Qwen-2.5-7B-Base, SFT, and SOC across different beam widths. SOC consistently outperforms Base model and SFT across various beam widths on both benchmarks, showcasing better performance with less computation demand for the inference.

the inference process. By mitigating the impact of value function estimation error with optimism correction, SOC improves the performance of beam search and significantly alleviates this issue.

5.4 Ablation

Beam Search Width To investigate the impact of the beam search width on performance and how SOC corrects the over-optimism in various searching spaces, we compare the Base, SFT, and SOC across different beam widths. The results are shown in Figure 3. We observe that: (1) on GSM8K, SFT fails to improve the performance of the Base model on beam search, instead, slightly degrading it. In contrast, SOC significantly enhances performance, highlighting its superiority. (2) on MATH-500, Base model has a limited performance compared to its accuracy of greedy decoding (54.6%) as shown in Table 1, demonstrating the harmful effect of the over-optimism problem on beam search. (3) on both benchmarks, SOC consistently outperforms Base model and SFT across beam widths ranging from small to large, validating that after optimism correction, the correct reasoning candidate consequences are favored in the beam search process even with small searching space. The results exhibit that, with supervised optimism correction, beam search can find better responses in fewer search branches, thereby reducing the search space and inference cost.

Impact of λ In the main experiments, we set the hyperparameter λ in Equation (17) to 0.2 in the results of Table 1. To further analyze the impact of λ on the performance of SOC, we conduct an ablation study by varying λ and evaluating its effect on the performance of SOC. The results are shown in Table 2.

We range λ in `{0.1, 0.2, 0.3}` for 7B model with DATA-β and DATA-α, and 3B model with DATA-α, to analyze how λ affects the performance of SOC for different datasets and models. In general, we find that the performance is not sensitive to the value of λ. Comparing 7B model with DATA-β and DATA-α, we find that SOC is more effective and robust on DATA-α than DATA-β, which can be due to the larger (4times) dataset size of DATA-α. On DATA-α, 7B model gains more improvement than 3B model for all the λ values and for both the best performance is achieved with λ=0.1.

We observe that SOC achieves the best performance when λ is set to 0.2, which is consistent across different base models and datasets. This indicates that a moderate value of λ effectively balances the influence of the auxiliary V loss, leading to optimal performance improvements.

6 Related Work

6.1 RL for LLM

Previous studies have demonstrated that RL enhances the performance of LLMs by optimizing reward feedback. These approaches typically involve RL from Human Feedback (RLHF) (Stiennon et al., 2020; Rafailov et al., 2024b; Liu et al., 2025b) to align LLMs with human preference, self-correction (Kumar et al., 2024), or direct fine-tuning for reasoning ability from the base model (Guo et al., 2025). These techniques enable LLMs to generate more aligned, accurate, and coherent responses. While our work follows a similar analysis, particularly in the context of offline RL (Kumar et al., 2020; Lyu et al., 2022; Qing et al., 2024; Zhang et al., 2023a; Wen et al., 2024a,b), we focus on the overestimation problem during SFT. We draw inspiration from offline RL to examine this problem within the SFT stage, offering a novel perspective on how overestimation impacts the optimization process.

6.2 Search-based methods of LLMs

Search-based methods during test-time computation are crucial for enabling models to improve their output quality (Snell et al., 2024; Liu et al., 2025a). These methods include beam search (Snell et al., 2024), Best-of-N (BoN) sampling (Brown et al., 2024), and lookahead-search methods, like MCTS (Zhang et al.; Xie et al., 2024a), generate sequences of k steps and evaluate which paths to retain for further exploration. Among these methods, beam search is a simple and widely used search

Table 2: Results (accuracy % of pass@1) on mathematical reasoning tasks. We report the results of each model sampling with beam search (beam width of 5), to show the impact of different λ on the performance of SOC.

MODEL	GSM8K	MATH-500	GAOKAO-EN	AVERAGE
Qwen-2.5-7B-Base	58.7	8.8	33.0	33.5
+SFT(DATA-β)	79.9	41.0	44.9	55.3
+SFT(DATA-β)+SOC(λ=0.1)	79.7	41.4	44.7	55.3(+0.0)
+SFT(DATA-β)+SOC(λ=0.2)	83.7	41.4	45.7	56.9(+1.6)
+SFT(DATA-β)+SOC(λ=0.3)	81.4	41.2	46.0	56.2(+0.9)
+SFT(DATA-α)	54.7	41.4	28.6	41.6
+SFT(DATA-α)+SOC(λ=0.1)	86.6	60.8	52.7	66.7(+25.1)
+SFT(DATA-α)+SOC(λ=0.2)	87.9	60.4	51.4	66.6(+25.0)
+SFT(DATA-α)+SOC(λ=0.3)	87.2	57.6	52.2	65.7(+24.1)
Qwen2.5-3B-Base	51.5	40.8	31.4	41.2
+SFT(DATA-α)	79.2	49.6	34.8	54.5
+SFT(DATA-α)+SOC(λ=0.1)	82.6	55.4	47.3	61.8(+7.3)
+SFT(DATA-α)+SOC(λ=0.2)	83.5	54.4	46.5	61.5(+7.0)
+SFT(DATA-α)+SOC(λ=0.3)	84.0	51.4	49.4	61.6(+7.1)

method without extra reward models or verifiers. Li (2024) first investigates the positive-incentive effects of noise and pioneers a series of studies on leveraging structured randomness to support task performance, offering valuable insights for test-time exploration strategies. Arora et al. (2022) provides a theoretical and empirical analysis of exposure bias in text generation and Wang et al. (2022) demonstrates that self-consistency can improve the performance of LLMs, using random sampling (noise) to explore multiple reasoning trajectories and then aggregating them. In this work, we examine how search procedures, particularly beam search, may suffer from over-optimism due to implicit maximization of Q-values during inference. Our work focuses on refining beam search techniques by addressing the over-optimism problem, which can lead to inflated Q-value estimates and amplified reasoning errors, particularly in long-horizon or sparse-reward settings.

7 Conclusion

In this paper, we formulate LLMs as token-level MDPs and establish a theoretical equivalence between SFT and offline RL, where LLMs implicitly learn a Q-function. We further show that beam search, a widely used decoding method, relies on this implicit Q-function but suffers from over-optimism due to value estimation errors. Based on that, we propose SOC, a simple auxiliary loss applied during SFT. Despite its simplicity, SOC is theoretically proven to effectively correct optimism, leading to more reliable guidance in inference time. Extensive experiments on mathematical reasoning benchmarks demonstrate that SOC significantly enhances reasoning performance across state-of-the-art open-source models.

Limitations

Although we conduct a comprehensive analysis of the over-optimism problem and the proposed method SOC, certain limitations remain, along with potential future research directions to explore: (1) While this work mainly focuses on the over-optimism problem of LLMs, it is valuable to investigate the issue of multi-modal models such as Visual Language Models (VLMs). (2) Investigating whether other search-based methods of LLMs encounter this issue is another direction and important to the development of test-time computation.

Ethical Considerations

We believe this work contributes to the development of LLMs in the field of NLP. It is worth mentioning that all the experiments are conducted using open-source models and datasets, ensuring no potential social concerns.

Acknowledgments

This research is supported by the RIE2025 Industry Alignment Fund – Industry Collaboration Projects (IAF-ICP) (Award I2301E0026), administered by A*STAR, as well as supported by Alibaba Group and NTU Singapore through Alibaba-NTU Global e-Sustainability CorpLab (ANGEL).

References

Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Chi Kit Cheung. 2022. Why exposure bias matters: An imitation learning perspective of error accumulation in language generation. *arXiv preprint arXiv:2204.01171*.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Yifu Ding, Wentao Jiang, Shunyu Liu, Yongcheng Jing, Jinyang Guo, Yingjie Wang, Jing Zhang, Zengmao Wang, Ziwei Liu, Bo Du, et al. 2025. Dynamic parallel tree search for efficient llm reasoning. *arXiv preprint arXiv:2502.16235*.

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

Han Guo, Bowen Tan, Zhengzhong Liu, Eric P Xing, and Zhiting Hu. 2021. Efficient (soft) q-learning for text generation with limited good data. *arXiv preprint arXiv:2106.07704*.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. 2024. A survey on large language models for code generation. *arXiv preprint arXiv:2406.00515*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. 2021. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, et al. 2024. Training language models to self-correct via reinforcement learning. *arXiv preprint arXiv:2409.12917*.

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

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. 2022. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. *Advances in Neural Information Processing Systems*, 35:21314–21328.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*.

Xuelong Li. 2024. Positive-incentive noise. *IEEE Transactions on Neural Networks and Learning Systems*, 35(6):8708–8714.

Minpeng Liao, Chengxi Li, Wei Luo, Wu Jing, and Kai Fan. 2024. MARIO: MAth reasoning with code interpreter output - a reproducible pipeline. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 905–924, Bangkok, Thailand. Association for Computational Linguistics.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Jiate Liu, Yiqin Zhu, Kaiwen Xiao, Qiang Fu, Xiao Han, Wei Yang, and Deheng Ye. 2023. Rltf: Reinforcement learning from unit test feedback. *arXiv preprint arXiv:2307.04349*.

Runze Liu, Junqi Gao, Jian Zhao, Kaiyan Zhang, Xiu Li, Biqing Qi, Wanli Ouyang, and Bowen Zhou. 2025a. Can 1b llm surpass 405b llm? rethinking compute-optimal test-time scaling. *arXiv preprint arXiv:2502.06703*.

Shunyu Liu, Wenkai Fang, Zetian Hu, Junjie Zhang, Yang Zhou, Kongcheng Zhang, Rongcheng Tu, Ting-En Lin, Fei Huang, Mingli Song, and Dacheng Tao. 2025b. A survey of direct preference optimization. *arXiv preprint arXiv:2503.11701*.

Jiafei Lyu, Xiaoteng Ma, Xiu Li, and Zongqing Lu. 2022. Mildly conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:1711–1724.

Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. 2022. Codegen: An open large language model for code with multi-turn program synthesis. *arXiv preprint arXiv:2203.13474*.

Damian Pascual, Beni Egressy, Florian Bolli, and Roger Wattenhofer. 2020. Directed beam search: Plug-and-play lexically constrained language generation. *arXiv preprint arXiv:2012.15416*.

Yunpeng Qing, Shunyu Liu, Jingyuan Cong, Kaixuan Chen, Yihe Zhou, and Mingli Song. 2024. A2po: Towards effective offline reinforcement learning from an advantage-aware perspective. In *Advances in Neural Information Processing Systems*, volume 37, pages 29064–29090.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. 2024a. From r to q^*: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024b. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

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

Hao Sun, Xiao Liu, Yeyun Gong, Yan Zhang, Daxin Jiang, Linjun Yang, and Nan Duan. 2023. Allies: Prompting large language model with beam search. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3794–3805, Singapore. Association for Computational Linguistics.

Richard S. Sutton and Andrew G. Barto. 1998. *Introduction to Reinforcement Learning*, 1st edition. MIT Press, Cambridge, MA, USA.

Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, Yongbin Li, Zhi Jin, Fei Huang, Dacheng Tao, and Jingren Zhou. 2024. A survey on self-evolution of large language models. *arXiv preprint arXiv:2404.14387*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7.

Sebastian Thrun and Anton Schwartz. 2014. Issues in using function approximation for reinforcement learning. In *Proceedings of the 1993 connectionist models summer school*, pages 255–263. Psychology Press.

Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Sheng Wang, Zihao Zhao, Xi Ouyang, Qian Wang, and Dinggang Shen. 2023. Chatcad: Interactive computer-aided diagnosis on medical image using large language models. *arXiv preprint arXiv:2302.07257*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

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

Xiaoyu Wen, Chenjia Bai, Kang Xu, Xudong Yu, Yang Zhang, Xuelong Li, and Zhen Wang. 2024a. Contrastive representation for data filtering in cross-domain offline reinforcement learning. *arXiv preprint arXiv:2405.06192*.

Xiaoyu Wen, Xudong Yu, Rui Yang, Haoyuan Chen, Chenjia Bai, and Zhen Wang. 2024b. Towards robust offline-to-online reinforcement learning via uncertainty and smoothness. *Journal of Artificial Intelligence Research*, 81:481–509.

Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P Lillicrap, Kenji Kawaguchi, and Michael Shieh. 2024a. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*.

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, James Xu Zhao, Min-Yen Kan, Junxian He, and Michael Xie. 2024b. Self-evaluation guided beam search for reasoning. *Advances in Neural Information Processing Systems*, 36.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024a. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. 2024b. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Huanjin Yao, Jiaxing Huang, Wenhao Wu, Jingyi Zhang, Yibo Wang, Shunyu Liu, Yingjie Wang, Yuxin Song, Haocheng Feng, Li Shen, and Dacheng Tao. 2024. Mulberry: Empowering mllm with o1-like reasoning and reflection via collective monte carlo tree search. *arXiv preprint arXiv:2412.18319*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. Rest-mcts*: Llm self-training via process reward guided tree search, 2024a. *URL https://arxiv. org/abs/2406.03816*.

Jingyi Zhang, Jiaxing Huang, Huanjin Yao, Shunyu Liu, Xikun Zhang, Shijian Lu, and Dacheng Tao. 2025a. R1-vl: Learning to reason with multimodal large language models via step-wise group relative policy optimization. *arXiv preprint arXiv:2503.12937*.

Junjie Zhang, Jiafei Lyu, Xiaoteng Ma, Jiangpeng Yan, Jun Yang, Le Wan, and Xiu Li. 2023a. Uncertainty-driven trajectory truncation for model-based offline reinforcement learning. *arXiv preprint arXiv:2304.04660*.

Kongcheng Zhang, Qi Yao, Baisheng Lai, Jiaxing Huang, Wenkai Fang, Dacheng Tao, Mingli Song, and Shunyu Liu. 2025b. Reasoning with reinforced functional token tuning. *arXiv preprint arXiv:2502.13389*.

Xinlu Zhang, Chenxin Tian, Xianjun Yang, Lichang Chen, Zekun Li, and Linda Ruth Petzold. 2023b. Alpacare: Instruction-tuned large language models for medical application. *arXiv preprint arXiv:2310.14558*.

Han Zhong, Guhao Feng, Wei Xiong, Xinle Cheng, Li Zhao, Di He, Jiang Bian, and Liwei Wang. 2024. Dpo meets ppo: Reinforced token optimization for rlhf. *arXiv preprint arXiv:2404.18922*.

Han Zhong, Wei Xiong, Sirui Zheng, Liwei Wang, Zhaoran Wang, Zhuoran Yang, and Tong Zhang. 2022. Gec: A unified framework for interactive decision making in mdp, pomdp, and beyond. *arXiv preprint arXiv:2211.01962*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Brian D Ziebart. 2010. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University.

A Proof of Theorem 4.1

Theorem A.1 (Contraction of Value Differences)**.** *Let $V_\theta(s)$ be the approximate value function of state s. Suppose the value function is always positive for any state s. If the objective in SFT includes a additional term in Equation (17), then after one step of gradient descent with learning rate $\alpha \in [0,1]$, the gap between adjacent states' values contracts, i.e.,*

$$|V'_\theta(s_t) - V'_\theta(s_{t+1})| \le |V_\theta(s_t) - V_\theta(s_{t+1})|, \quad \forall t,$$

where V' denotes the updated value function after one optimization step.

Proof. Recall that the auxiliary objective is to minimize $-\log V(s)$. After one step of gradient descent, the updated value function becomes $V'(s) = V(s) + \alpha \cdot \frac{1}{V(s)}$. Now, we can compute the difference between the updated values for any adjacent states s_t and s_{t+1}:

$$\begin{aligned}
&\left|V'(s_t) - V'(s_{t+1})\right| \\
&= \left|V(s_t) - V(s_{t+1}) + \alpha\left(\frac{1}{V(s_t)} - \frac{1}{V(s_{t+1})}\right)\right| \\
&\le \left|1 - \frac{\alpha}{V(s_t)V(s_{t+1})}\right| \cdot |V(s_t) - V(s_{t+1})| \\
&\le |V(s_t) - V(s_{t+1})| .
\end{aligned} \tag{22}$$

The last inequality holds since we assume the value function is always positive and learning rate $\alpha \in [0,1]$. Thus, we see that after one step of gradient descent, the gap between the updated value functions of adjacent states is no greater than the original gap. □

B The Derivation for Estimation Error

We provide a proof sketch for proof of Equation (16). We start from SFT objective, which is given by:

$$\mathcal{L}_{\text{SFT}} = \mathbb{E}_{s\sim\mathcal{D}}\left[-\sum_a \pi^*(a|s)\log\pi_\theta(a|s)\right]. \tag{23}$$

To compute its gradient, we first differentiate the softmax function:

$$\pi_\theta(a|s) = \frac{\exp\left(Q_\theta(a|s)\right)}{\sum_{a'}\exp\left(Q_\theta(a'|s)\right)}. \tag{24}$$

Taking the gradient with respect to the logits $Q_\theta(a|s)$, we obtain:

$$\frac{\partial\pi_\theta(a|s)}{\partial Q_\theta(a'|s)} = \pi_\theta(a|s)\left(\mathbb{I}[a = a'] - \pi_\theta(a'|s)\right). \tag{25}$$

Next, we differentiate the loss function:

$$\nabla_\theta\mathcal{L}_{\text{SFT}} = \mathbb{E}_{s\sim\mathcal{D}}\left[-\sum_a \pi^*(a|s)\frac{\partial\log\pi_\theta(a|s)}{\partial\theta}\right]. \tag{26}$$

Since

$$\frac{\partial\log\pi_\theta(a|s)}{\partial\theta} = \frac{1}{\pi_\theta(a|s)}\frac{\partial\pi_\theta(a|s)}{\partial\theta}, \tag{27}$$

we obtain:

$$-\sum_a \pi^*(a|s)\sum_{a'}\frac{1}{\pi_\theta(a|s)}\frac{\partial\pi_\theta(a|s)}{\partial Q_\theta(a'|s)}\frac{\partial Q_\theta(a'|s)}{\partial\theta}. \tag{28}$$

Substituting the gradient of $\pi_\theta(a|s)$:

$$-\textstyle\sum_a \pi^*(a|s)\sum_{a'}\left(\mathbb{I}[a=a'] - \pi_\theta(a'|s)\right)\frac{\partial Q_\theta(a'|s)}{\partial\theta}. \tag{29}$$

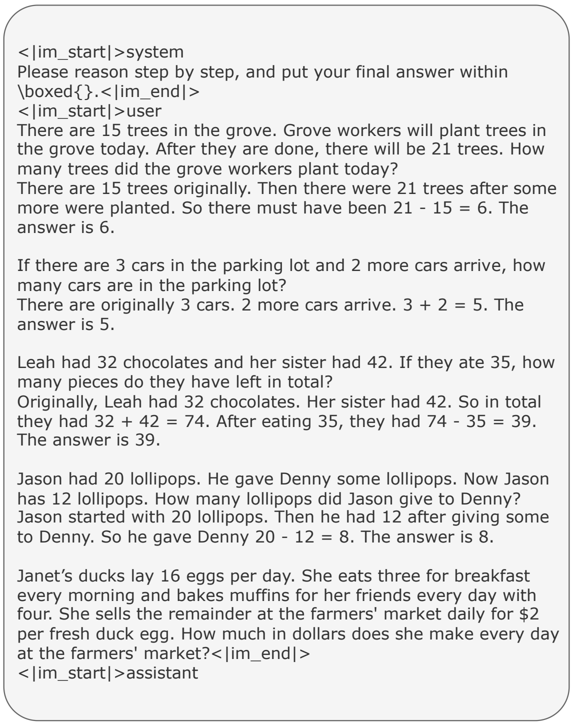
Rearranging the terms, we obtain:

$$\nabla_\theta\mathcal{L}_{\text{SFT}} = \mathbb{E}_{s\sim\mathcal{D}}\left[\textstyle\sum_a\left(\pi_\theta(a|s) - \pi^*(a|s)\right)\frac{\partial Q_\theta(a|s)}{\partial\theta}\right]. \tag{30}$$

We finish the proof by replacing π with normalized $\widetilde{Q}$, which is equivalent and unambiguous.

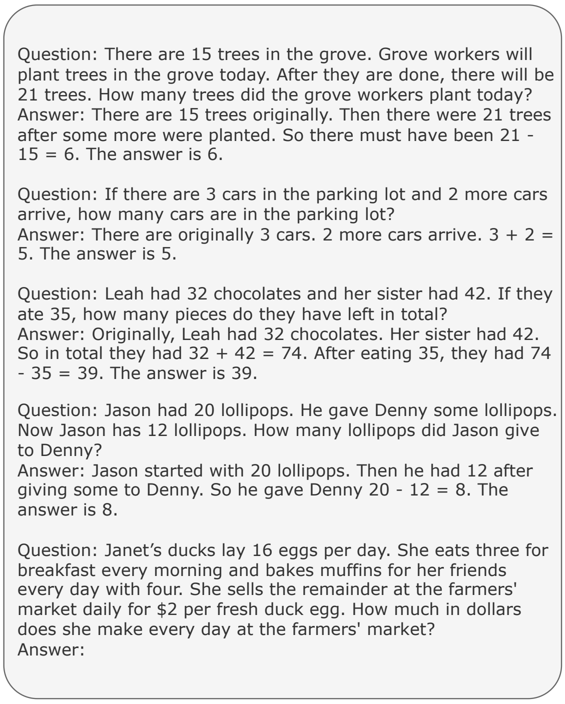
C Prompt for Evaluation

Here we provide the prompt for evaluation in our experiments. Following previous work (Yang et al., 2024b), we use CoT with few-shots demonstrations to prompt LLMs to solve mathematical problems. The two kinds of templates are shown in Figure 4 and Figure 5.



```
<|im_start|>system
Please reason step by step, and put your final answer within
\boxed{}.<|im_end|>
<|im_start|>user
There are 15 trees in the grove. Grove workers will plant trees in
the grove today. After they are done, there will be 21 trees. How
many trees did the grove workers plant today?
There are 15 trees originally. Then there were 21 trees after some
more were planted. So there must have been 21 - 15 = 6. The
answer is 6.

If there are 3 cars in the parking lot and 2 more cars arrive, how
many cars are in the parking lot?
There are originally 3 cars. 2 more cars arrive. 3 + 2 = 5. The
answer is 5.

Leah had 32 chocolates and her sister had 42. If they ate 35, how
many pieces do they have left in total?
Originally, Leah had 32 chocolates. Her sister had 42. So in total
they had 32 + 42 = 74. After eating 35, they had 74 - 35 = 39.
The answer is 39.

Jason had 20 lollipops. He gave Denny some lollipops. Now Jason
has 12 lollipops. How many lollipops did Jason give to Denny?
Jason started with 20 lollipops. Then he had 12 after giving some
to Denny. So he gave Denny 20 - 12 = 8. The answer is 8.

Janet’s ducks lay 16 eggs per day. She eats three for breakfast
every morning and bakes muffins for her friends every day with
four. She sells the remainder at the farmers' market daily for $2
per fresh duck egg. How much in dollars does she make every day
at the farmers' market?<|im_end|>
<|im_start|>assistant
```

Figure 4: Template 1 for prompt.



```
Question: There are 15 trees in the grove. Grove workers will
plant trees in the grove today. After they are done, there will be
21 trees. How many trees did the grove workers plant today?
Answer: There are 15 trees originally. Then there were 21 trees
after some more were planted. So there must have been 21 -
15 = 6. The answer is 6.

Question: If there are 3 cars in the parking lot and 2 more cars
arrive, how many cars are in the parking lot?
Answer: There are originally 3 cars. 2 more cars arrive. 3 + 2 =
5. The answer is 5.

Question: Leah had 32 chocolates and her sister had 42. If they
ate 35, how many pieces do they have left in total?
Answer: Originally, Leah had 32 chocolates. Her sister had 42.
So in total they had 32 + 42 = 74. After eating 35, they had 74
- 35 = 39. The answer is 39.

Question: Jason had 20 lollipops. He gave Denny some lollipops.
Now Jason has 12 lollipops. How many lollipops did Jason give
to Denny?
Answer: Jason started with 20 lollipops. Then he had 12 after
giving some to Denny. So he gave Denny 20 - 12 = 8. The
answer is 8.

Question: Janet’s ducks lay 16 eggs per day. She eats three for
breakfast every morning and bakes muffins for her friends
every day with four. She sells the remainder at the farmers'
market daily for $2 per fresh duck egg. How much in dollars
does she make every day at the farmers' market?
Answer:
```

Figure 5: Template 2 for prompt.