Direct Value Optimization: Improving Chain-of-Thought Reasoning in LLMs with Refined Values

Hongbo Zhang[1,2]*, Han Cui[1,2]*, Guangsheng Bao[1,2]*, Linyi Yang[4], Jun Wang[5], Yue Zhang[2,3]†
[1] Zhejiang University
[2] School of Engineering, Westlake University
[3] Institute of Advanced Technology, Westlake Institute for Advanced Study
[4] Southern University of Science and Technology
[5] University College London
{zhanghongbo,cuihan,baoguangsheng,zhangyue}@westlake.edu.cn
yanglinyiucd@gmail.com jun.wang@cs.ucl.ac.uk

Abstract

We introduce Direct Value Optimization (DVO), an innovative reinforcement learning framework for enhancing large language models in complex reasoning tasks. Unlike traditional methods relying on preference labels, DVO utilizes value signals at individual reasoning steps, optimizing models via a mean squared error loss. The key benefit of DVO lies in its fine-grained supervision, circumventing the need for labor-intensive human annotations. Target values within the DVO are estimated using either Monte Carlo Tree Search or an outcome value model. Our empirical analysis on both mathematical and commonsense reasoning tasks shows that DVO consistently outperforms existing offline preference optimization techniques, even with fewer training steps. These findings underscore the importance of value signals in advancing reasoning capabilities and highlight DVO as a superior methodology under scenarios lacking explicit human preference information. The code is available at https://github.com/StevenZHB/DVO.

1 Introduction

Large language models (LLMs) have demonstrated exceptional performance across a wide range of natural language processing (NLP) tasks (Yu et al., 2023b; Azerbayev et al., 2023; Reid et al., 2024; Chen et al., 2023; Achiam et al., 2023; Yang et al., 2024a). However, when addressing complex problems such as mathematical reasoning (Yue et al., 2023; Yu et al., 2023c), logical reasoning (Liu et al., 2023a; Teng et al., 2023), and multi-step planning problems (Hao et al., 2023), they can fall into incorrect reasoning paths due to errors in intermediate reasoning steps (Cobbe et al., 2021; Shen et al., 2021; Bao et al., 2025).

To address this issue, recent studies leverage reinforcement learning (RL) to optimize the reasoning process according to feedback derived from golden answers (Shao et al., 2024; Luo et al., 2023; Chen et al., 2024d; Kazemnejad et al., 2024), significantly enhancing the robustness. Among these, converting reward signals into preference labels has been a standard practice, where multiple responses are generated for one question and the optimal response reaching at the correct answer is labeled as the preferred response in contrast to others. Preference labels are then used to train models through a cross-entropy loss or a contrastive alternatives. For example, DPO (Rafailov et al., 2023), KTO (Ethayarajh et al., 2024), and RFT (Yuan et al., 2024, 2023) optimize language models based on preferences at the token level, while recent process supervision (Lightman et al., 2023), CPO (Zhang et al., 2024), Step-DPO (Lai et al., 2024), SVPO (Chen et al., 2024b) and Iterative Preference Learning (Xie et al., 2024) generate preferences and optimize LLMs at the step level.

Preference labels as training targets are simple and convenient for implementation. However, rewards from preference labels lack fine-grained signals and can fail to preserve critical value information (Liu et al., 2023b; Chen et al., 2024b). A key limitation of preference-based optimization is its reliance on pairwise comparisons, which only indicate relative preference between two choices rather than providing an absolute ranking of multiple steps. In contrast, stepwise value signals offer a continuous ranking, allowing the model to make more informed decisions beyond binary comparisons. By leveraging value signals, models can better understand reasoning dynamics, prioritizing steps that enhance clarity, reduce ambiguity, and improve multi-step reasoning efficiency.

To address this issue, we propose *Direct Value Optimization (DVO)*, a simple yet effective reinforcement learning (RL) framework for large language models (LLMs). As shown in Fig. 1, DVO estimates target values at each reasoning step and

*Equal contribution.
†Corresponding author.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 13203–13216
November 4-9, 2025

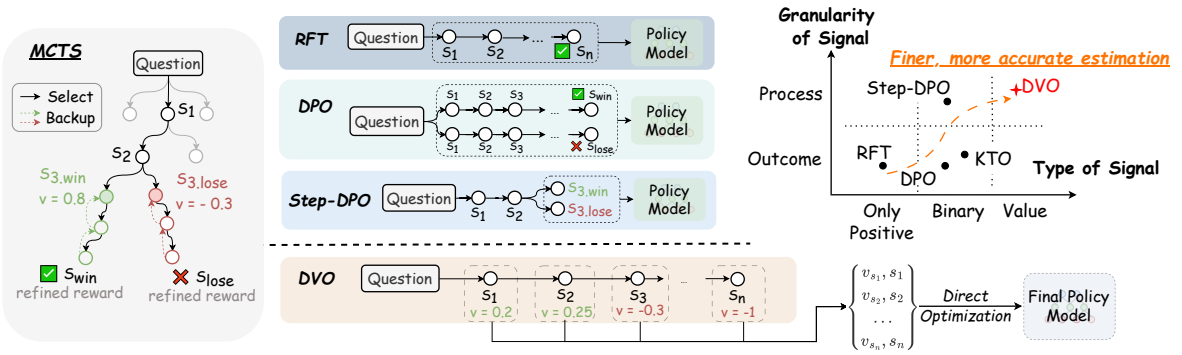


Figure 1: Overview of Direct Value Optimization. Compared with other self-improvement methods, DVO stands out by using MCTS to generate self-explored data and directly aligning the policy model with value estimations. This provides an efficient self-improvement framework that finely tunes the model to maximize total expected rewards.

aligns the model with these values using a mean squared error (MSE) loss. To make use of widely available outcome-supervised data, we infer step-wise value signals from the final outcome, providing process-level guidance for RL. The key distinction between DVO and other offline RL methods lies in its ability to optimize LLMs using value signals without requiring process-level human labeling, thereby eliminating the need for labor-intensive annotations while retaining fine-grained supervision. In DVO, target values can be estimated using various methods. Specifically, we explore two primary approaches: monte carlo tree search (MCTS) (Kocsis and Szepesvári, 2006; Coulom, 2006) and outcome value model (Yu et al., 2023a; Feng et al., 2023).

We conduct extensive experiments on both mathematical reasoning tasks and commonsense reasoning tasks, evaluating models with various sizes. Results show that DVO outperforms other offline preference optimization algorithms across multiple benchmarks under the same number of training steps. For example, a three rounds of DVO training improves Llama3-8B-Instruct's (Dubey et al., 2024) accuracy on GSM8K (Cobbe et al., 2021) from 74.6% to 80.6%, on MATH (Hendrycks et al., 2021) from 22.5% to 26.5%, and on AGIEval-Math (Zhong et al., 2023) from 23.5% to 27.9%. Finally, comprehensive analytical experiments highlight the critical role of value signals in reasoning tasks. Our code is available at <ANONYMOUS>.

2 Related Work

Improving Reasoning in LLMs. Existing studies improve the robustness of LLM reasoning at various stages. Some focus on the pre-training phase by curating pretraining datasets to enhance mathematical knowledge (Shao et al., 2024; Azerbayev et al., 2023), some address it during instruction fine-tuning using high-quality reasoning QA datasets (Mitra et al., 2023; Mukherjee et al., 2023; Yue et al., 2023; Yu et al., 2023c), while others tackle the reasoning challenge at the reinforcement learning stage by using human or AI feedback for optimization (Yuan et al., 2024; Lai et al., 2024). Additionally, several studies improve reasoning at the inference stage through prompting techniques that search for better reasoning paths in trees or graphs (Yao et al., 2024; Besta et al., 2024; Wang et al., 2023a; Yang et al., 2024b; Zheng et al., 2023). Our work aligns most closely with methods that focus on the reinforcement learning stage but differs by improving LLMs by using value signals.

Reinforcement Learning in LLMs. Reinforcement learning is widely used in LLM post-training to learn implicit rewards from human feedback (Ouyang et al., 2022; Dubey et al., 2024; Yang et al., 2024a). The standard pipeline trains a reward model and updates the policy via PPO (Schulman et al., 2017b). DPO (Rafailov et al., 2023) shows that the log ratio between optimal policy and reference models expresses the corresponding reward model, allowing direct policy updates based on the preference data. Rafailov et al. (2024) further derived the DPO objective, validating its ability to learn token-level rewards, while subsequent works refine DPO for practical applications (Ethayarajh et al., 2024; Chen et al., 2024c; Azar et al., 2024). Our approach extends DPO by aligning the policy model with value estimation, eliminating the need for pairwise data and leveraging value signals for enhanced reasoning. While recent work like DQO (Liu et al., 2024) and OREO (Wang et al.,

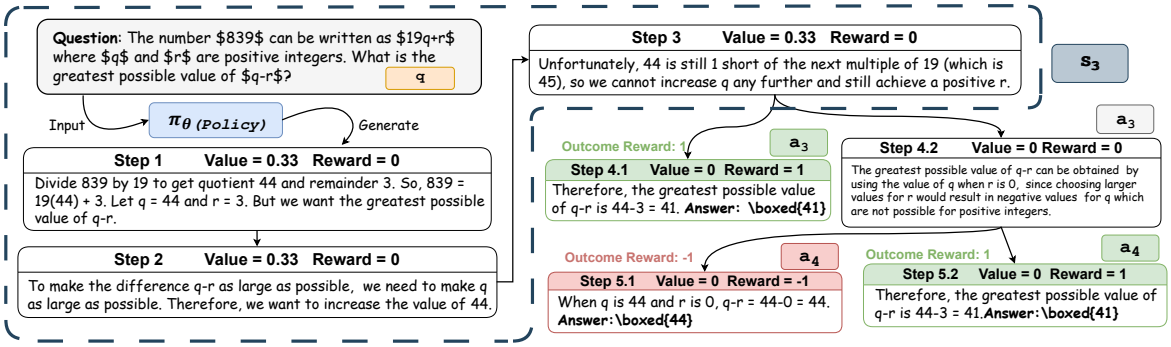


Figure 2: Illustration of Step-by-step reasoning process, each node represents a step with its corresponding reward and value estimation.

2024) also aims to improve LLM reasoning within the value-based framework, they differ from our approach in that they require training a separate value model alongside the policy model. In contrast, our method is more closely aligned with DPO, requiring only the policy model and a reference model during training.

Tree-search Guided Reasoning in LLMs. Tree search has recently emerged as a key technique in LLM reasoning. Prior work has used prompt engineering to estimate expected returns for reasoning steps, guiding models toward optimal answers (Yao et al., 2024; Feng et al., 2023; Besta et al., 2024). Subsequent studies employed tree search to generate training data for RL or fine-tuning. Xie et al. (2024) and Chen et al. (2024b) use value estimates from tree search to create step-level preference pairs, while Feng et al. (2023) and Chen et al. (2024a) train value models from these estimates for guided decoding. Our work also leverages MCTS, but unlike Feng et al. (2023) and Chen et al. (2024a) who focus on improving the inference stage, we directly train the policy model using MCTS value estimates. Additionally, compared to Xie et al. (2024) and Chen et al. (2024b), we make use of value estimates instead of preference, obtaining stronger results.

3 Baseline: Direct Preference Optimization

Direct Preference Optimization (DPO) (Rafailov et al., 2023) was designed to align LLMs with human preference labels. Recent work (Yuan et al., 2023; Ethayarajh et al., 2024; Chen et al., 2024c) has extended DPO to optimize LLMs with reward signals by converting them into preference labels through specific algorithms. While DPO was initially developed for the contextual bandit setting, it can be validated within token-level MDP (Rafailov et al., 2024). In the RLHF pipeline, we first train a reward model based on the Bradley-Terry framework (Bradley and Terry, 1952), which is well-suited for modeling implicit human preferences:

$$P_{\mathrm{BT}}(y_1 \succ y_2 \mid x) = \frac{\exp(r(x, y_1))}{\exp(r(x, y_1)) + \exp(r(x, y_2))}, \quad (1)$$

where x is the prompt, y_1, y_2 are two completions. The optimal policy, Q-function, and V-function learned from a reward function R are denoted as π^*, Q^*, and V^*, respectively. satisfying the relation in Eq. (4). Using π^*, Q^*, and V^*, the accumulated step-level reward is:

$$\begin{aligned} R(x, y) &= \sum_{t=1}^{T-1} (Q^*(\mathbf{s}_t, \mathbf{a}_t) - V^*(\mathbf{s}_{t+1})) \\ &= V^*(\mathbf{s}_0) - V^*(\mathbf{s}_T) + \beta \log \pi^*(y \mid x) \\ &= V^*(\mathbf{s}_0) + \beta \log \pi^*(y \mid x) \end{aligned}$$

where we assume $V^*(s_T) = 0$. To prevent the model from deviating excessively from its prior behavior during training, the reward function is:

$$R(a_t \mid s_t) = \beta \log \pi_{\mathrm{ref}}(a_t \mid s_t) + r(a_t \mid s_t),$$

which leads to the reward model as:

$$r(x, y) = \beta \log \frac{\pi^*(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} + V^*(\mathbf{s}_0).$$

This establishes a connection between the learned policy and the reward model, enabling direct optimization of the policy without the need for explicit reward modeling. Substituting the regularized reward into the Bradley-Terry framework yields the DPO objective:

$$\begin{aligned} \mathcal{L}_{\mathrm{DPO}} = -\mathbb{E}_{(x, y_w, y_l)} \Bigg[\log \sigma \Bigg(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\mathrm{ref}}(y_w \mid x)} \\ - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\mathrm{ref}}(y_l \mid x)} \Bigg) \Bigg], \end{aligned} \quad (2)$$

where σ denotes the sigmoid function, and y_w, y_l represent winning and losing responses respectively. The baseline term $V^*(\mathbf{s}_0)$ cancels out in pairwise comparisons, eliminating the need for explicit value estimation ($\mathbf{s}_0$ is the same for both y_w and y_l). This derivation is based on a token-level MDP but can also be applied to the step-level MDP introduced later in this work.

4 Direct Value Optimization

We derive a framework that optimize reasoning of LLMs using trajectories along with their estimated stepwise values by extending soft Q-learning. To our knowledge, this is the first work to directly optimize LLMs using estimated value signals within a Q-learning framework.

4.1 Problem Definition: A Stepwise MDP

As Fig. 2 illustrates, we formulate the reasoning problem as a stepwise Markov Decision Process (step MDP), where a whole reasoning process is split into several small but coherent reasoning steps, each representing a link in the chain of thoughts.

Formally, at each step t, the policy model is in a state $\mathbf{s}_t$, which consists of the current question q (sampled from a distribution $\mathcal{D}$) and a sequence of prior steps $[y_1, y_2, \ldots, y_{t-1}]$, denoted as $\mathbf{s}_t = \{q, y^{<t}\}$. The policy model π_θ selects the next action $\mathbf{a}_t$ from the set of possible steps according to the probability distribution $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$, guiding the reasoning process forward. The goal of the model is to produce a sequence of steps that ultimately leads to the final answer.

To encourage the model to perform robust reasoning steps that lead to the correct outcome, we define a reward function $r(\mathbf{s}_t, \mathbf{a}_t)$. The reward evaluates the quality of each reasoning step, ensuring that the model focuses on generating logical and accurate transitions. In this context, the reasoning process is deterministic: the next state $\mathbf{s}_{t+1}$ is fully determined by the current state $\mathbf{s}_t$ and the selected action $\mathbf{a}_t$. As a result, we omit explicit references to transition probabilities $P(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)$ in subsequent equations for simplicity.

4.2 LLM as Soft Q-Function

Maximum entropy reinforcement learning introduces an entropy term to encourage exploration, balancing reward maximization and stochasticity for robust decision-making (Rafailov et al., 2024; Ziebart, 2010; Haarnoja et al., 2017). In this framework, the Soft Q-function and Soft V-function are defined as follows:

$$\begin{aligned} Q_{\text{soft}}(\mathbf{s}_t, \mathbf{a}_t) &= R(\mathbf{s}_t, \mathbf{a}_t) + V_{\text{soft}}(\mathbf{s}_{t+1}), \\ V_{\text{soft}}(\mathbf{s}_t) &= \beta \log \int_{\mathcal{A}} \exp(\frac{1}{\beta} Q_{\text{soft}}(\mathbf{s}_t, \mathbf{a}'))d\mathbf{a}', \end{aligned} \tag{3}$$

where $\mathcal{A}$ is the action space, and β is a temperature parameter controlling the trade-off between entropy and reward. The optimal policy π^* can be expressed in terms of the optimal Soft Q-function Q^*_{soft} and optimal Soft V-function V^*_{soft}:

$$\pi^*(\mathbf{a}_t \mid \mathbf{s}_t) = \exp\left(\frac{Q^*_{\text{soft}}(\mathbf{s}_t, \mathbf{a}_t) - V^*_{\text{soft}}(\mathbf{s}_t)}{\beta}\right). \tag{4}$$

Building on this framework, Rafailov et al. (2024) formally proposed that an LM can be interpreted as an optimal soft Q-functions.

Proposition 1. *(Proof in Appendix A) In general maximum entropy reinforcement learning setting, a language model parameterized by π_θ can be seen as an optimal soft Q-function under some reward.*

Thus, the corresponding optimal Q-function can be presented by π_θ:

$$Q^\theta_{\text{soft}}(\mathbf{s}_t, \mathbf{a}_t) = \beta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t) + V^\theta_{\text{soft}}(\mathbf{s}_t). \tag{5}$$

4.3 The DVO Objective

Different from DPO, which optimizes language models at the response level using cross-entropy loss, we treat the policy as a Q-function and optimize it with Mean Squared Error (MSE) loss based on value estimations.

Specifically, following Haarnoja et al. (2017); Schulman et al. (2017a), the Q-function can be optimized by using soft Q-learning:

$$J_Q(\theta) = \mathbb{E}_{t, \mathbf{s}_t, \mathbf{a}_t}\left[\frac{1}{2}\left(Q^\theta_{\text{soft}}(\mathbf{s}_t, \mathbf{a}_t) - y_t\right)^2\right], \tag{6}$$

where y_t is the target Q-value. In one-step Q-learning, the target Q-value is computed as:

$$y_t = R(\mathbf{s}_t, \mathbf{a}_t) + \hat{V}^\theta_{\text{soft}}(\mathbf{s}_{t+1}), \tag{7}$$

where $\hat{V}^\theta_{\text{soft}}(\mathbf{s}_{t+1})$ is the target V-function of $\mathbf{s}_{t+1}$. Substituting Eq. 5 and Eq. 7 into the objective, we obtain:

$$\begin{aligned} J_Q(\theta) = \mathbb{E}_{t, \mathbf{s}_t, \mathbf{a}_t}[\,&\frac{1}{2}\Big(\beta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t) + V^\theta_{\text{soft}}(\mathbf{s}_t) \\ &- R(\mathbf{s}_t, \mathbf{a}_t) - \hat{V}^\theta_{\text{soft}}(\mathbf{s}_{t+1})\Big)^2\Big] \end{aligned} \tag{8}$$

Similar as DPO, the reward function $R(\mathbf{s}_t, \mathbf{a}_t)$ includes both the actual reward $r(\mathbf{s}_t, \mathbf{a}_t)$ and a KL penalty term:

$$R(\mathbf{s}_t, \mathbf{a}_t) = \beta \log \pi_{\text{ref}}(\mathbf{a}_t \mid \mathbf{s}_t) + r(\mathbf{s}_t, \mathbf{a}_t), \tag{9}$$

where π_{ref} is a reference policy. To optimize the parameters θ, we substitute the estimated value function $\hat{V}^{\theta}_{\text{soft}}(s_t)$ for $V^{\theta}_{\text{soft}}(s_t)$. This leads to the final objective:

$$J_Q(\theta) = \mathbb{E}_{t,\mathbf{s}_t,\mathbf{a}_t}[\frac{1}{2}\Bigg(\beta \log \frac{\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)}{\pi_{\text{ref}}(\mathbf{a}_t \mid \mathbf{s}_t)} - \left(\hat{V}^{\theta}_{\text{soft}}(\mathbf{s}_{t+1}) + r(\mathbf{s}_t, \mathbf{a}_t) - \hat{V}^{\theta}_{\text{soft}}(\mathbf{s}_t)\right)\Bigg)^2]. \tag{10}$$

As a result, we can now directly align a language model with step-level value estimation, which is derived from outcome reward signals without requiring human labeling.

Iterative Training. This approach can be naturally extended into an iterative process. In each iteration, the latest policy model is used to generate new reasoning paths and corresponding value estimates, which are then used for further training.

4.4 Target Value Estimation

We investigate two representative methods for estimating target values for DVO training: MC estimation with a tree search algorithm (Kocsis and Szepesvári, 2006; Coulom, 2006), which directly approximates the value through simulated rollouts, and an outcome value model (Yu et al., 2023a; Feng et al., 2023), which learns to predict the value based on the observed outcomes and states.

4.4.1 Estimation Using MCTS

We use Monte Carlo Tree Search (MCTS) to sample reasoning paths and estimate process value functions. Known for its success in AlphaZero and AlphaGo (Silver et al., 2017, 2016), MCTS effectively balances exploration and exploitation while providing accurate state value estimates. It has also been widely adopted in complex reasoning tasks for data generation (Xie et al., 2024; Chen et al., 2024a; Feng et al., 2023; Chen et al., 2024b). In this framework, we modify MCTS to collect training data for the policy model π_θ, incorporating the following key steps:

Selection: Starting from the root node, child nodes are recursively selected using the PUCT algorithm (Rosin, 2011) until an expandable leaf node is reached. The next node is chosen based on a trade-off between exploration and exploitation:

$$s' = \arg\max_{\mathbf{a}}[Q(\mathbf{s}, \mathbf{a}) + c \cdot \pi_\theta(\mathbf{a} \mid \mathbf{s}) \frac{\sqrt{N(\mathbf{s})}}{1 + N(\mathbf{s}')}].$$

Here, $Q(\mathbf{s}, \mathbf{a})$ represents the Q value estimate from the tree search, $N(\mathbf{s})$ is the visit count of node $\mathbf{s}$, and c is a hyperparameter that regulates the trade-off between exploration and exploitation.

Expansion: The selected node is expanded by adding the top-k most probable actions and their corresponding next states as child nodes.

Evaluation & Backup: Leaf nodes are assigned a KL-regularized reward (Eq. 9). During soft value function estimation, the entropy term from the policy model cancels out the KL regularization term, allowing the use of the original reward for sampling. Terminal node results are propagated back to the root node, updating visit counts and cumulative rewards:

$$N(s', a) \leftarrow N(s, a) + 1$$
$$V(s) \leftarrow \frac{\sum_a N(s')(r(s, a) + V(s'))}{\sum_a N(s')}$$

4.4.2 Estimation Using a Value Model

Outcome value models are widely used in inference-time tree search, providing value guidance for each step in the search process (Yu et al., 2023a; Wang et al., 2023b). We leverage a value model to assist in training the policy model π_θ. During value model training, we minimize a mean squared error loss to directly model the outcome reward:

$$L_V(\phi) = \mathbb{E}_{(s_i^{(n)}, r_i) \in D}\left[\frac{1}{n}\sum_{t=1}^{n} \|V_\phi(\mathbf{s}_t) - r_i\|_2^2\right],$$

where $s_i^{(n)}$ represents a trajectory of n states, r_i is the associated outcome reward, and $V\phi(\mathbf{s}_t)$ is the predicted value for $\mathbf{s}_t$ by the value model parameterized by ϕ. The data is defined as $D = \{(s_i^{(n)}, r_i)\}$, which is responses generated from π_θ. Once trained with D, the value model provides target values to update the policy model π_θ. Notably, it remains fixed during policy optimization and is not updated alongside π_θ.

5 Experimental Settings

Datasets. We validate our approach on both math word problems and commonsense reasoning tasks. For math word problems, we use GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) as in-domain test sets. The former contains over 1,300 grade school math problems, while the latter includes 5,000 problems spanning five difficulty levels. AGIEval-Math (Zhong et al., 2023) is used as an out-of-distribution (OOD) test set. For training, we take MetaMath (Yu et al., 2023c), a dataset

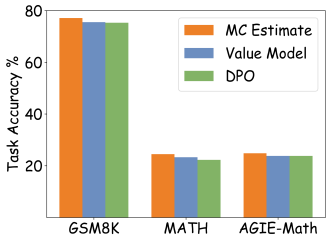


Figure 3: Different value estimation in DVO.

with 395K math problems, solutions, and correct answers. In our training process, only the correct answers are used, and solutions are excluded. During training, we sample 10,000 problems per round.

Models. We employ three models as our backbones: two state-of-the-art general language model in different size, Llama3-8B-Instruct, and Llama3-70B-Instruct (Dubey et al., 2024), and a specialized math-focused language model, DeepSeek-Math-Instruct-7B (Shao et al., 2024).

Implementation Details. Before DVO training, we fine-tune the model backbones to ensure proper formatting of reasoning steps, with "\n\n" used as a delimiter between steps and "Final Answer:" marking the conclusion. Our approach focuses on using MCTS for target value collection. All models are optimized over three rounds of DVO training. In each round, we perform a modified MCTS for each problem, where each node represents a reasoning step generated by the model, separated by "\n\n" The tree search is configured with a maximum of 80 iterations, a breadth limit of 5, and a depth limit of 8. From the search tree, we select four positive and four negative examples for policy updates. During evaluation, all assessments are conducted in a zero-shot setting. Details of the fine-tuning process and evaluation setting can be found in App. B.

Baselines. We compare our methods with several baselines which leverage self-generated data to boost the reasoning performance of LLMs, which are categorized based on how they train the policy model: (i) DPO (Rafailov et al., 2023; Yuan et al., 2024) samples positive and negative paths from the results of MCTS to construct pairs and use DPO to train the policy model at the solution level. (ii) Step-DPO (Zhang et al., 2024; Lai et al., 2024; Setlur et al., 2024) is a step-level DPO algorithm, aligning preferences at a finer granularity by constructing pair comparisons at the step level and training the policy model using DPO. (iii) KTO (Ethayarajh et al., 2024) is an extension of DPO, which trains the policy model without relying on paired data. (iv) RFT (Chen et al., 2024a; Feng et al., 2023; Yuan et al., 2023) directly samples correct paths from the tree and use supervised fine-tuning to train the policy model. In our comparative experiments, we use the same trees in the first round to ensure consistency across methods. Further details are provided in App. C.

Model	Size	In-domain		Out-of-domain
		GSM8K	MATH	AGIE-Math
DeepSeek-Math-7B-Ins.†	7B	81.4	44.4	43.4
+ RFT	7B	81.0	44.1	43.5
+ DPO	7B	82.1	46.0	44.8
+ KTO	7B	83.0	44.0	43.2
+ Step-DPO	7B	82.0	45.7	43.3
+ DVO (Ours)	7B	**85.2**	**47.6**	**46.3**
Llama-3-8B-Instruct†	8B	74.6	22.5	23.5
+ RFT	8B	75.6	23.4	23.9
+ DPO	8B	75.3	22.3	23.8
+ KTO	8B	76.0	24.1	24.3
+ Step-DPO	8B	76.6	23.3	23.5
+ DVO (Ours)	8B	**80.6**	**26.5**	**27.9**
Llama-3-70B-Instruct†	70B	85.4	34.9	33.8
+ RFT	70B	86.3	34.8	35.2
+ DPO	70B	82.7	32.6	31.5
+ KTO	70B	86.5	35.8	35.5
+ Step-DPO	70B	87.7	36.7	35.0
+ DVO (Ours)	70B	**90.7**	**40.3**	**38.9**

Table 1: Results on math reasoning tasks. † indicates that the model has been fine-tuned to follow step-by-step pattern. Highest performance results in comparison are highlighted in **bold**, and the second-high results are underlined for clarity.

6 Results

6.1 Target Value Estimation

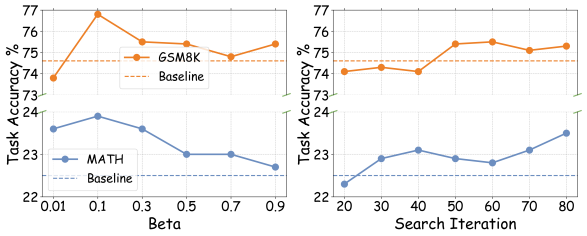
We first compare the two options for target value estimation: MC estimation and outcome value model. Specifically, we initialize the value network with Llama-3-8B-Instruct, following the OVM approach (Yu et al., 2023a) to train the network, and use it to label target values for DVO training data. The results, shown in Fig. 3, indicate that the MC estimation achieves better performance while avoiding the additional computational overhead required to train a separate value network. Therefore, all subsequent experiments are performed using MCTS for value estimation.

6.2 Main Results

We compare DVO with baseline methods on math reasoning tasks. As shown in Tab. 1, DVO consistently delivers stable gains across various model backbones and significantly outperforms DPO and its alternatives KTO and Step-DPO. Specifically, DVO improves DeepSeek-Math from 83.0% (KTO)

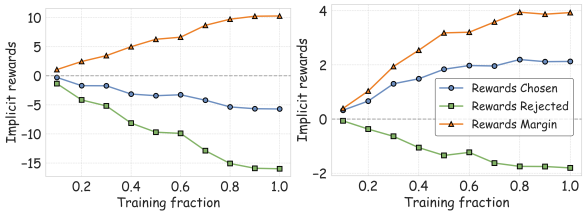
Model	In-domain			Out-of-domain
	OBQA	CSQA	ARC-C	SciQ
Llama-3-8B-Instruct†	81.2	74.1	79.7	90.7
+ RFT	79.6	77.0	73.9	87.0
+ DPO	82.8	78.7	80.9	90.6
+ KTO	83.4	79.0	81.4	91.5
+ Step-DPO	82.2	79.0	**82.0**	91.6
+ DVO (Ours)	**84.2**	**79.3**	81.7	**92.4**

Table 2: Results on commonsense reasoning tasks. Highest results are highlighted in **bold**.



(a) (b)

Figure 4: Ablation study on hyperparameters. The left figure demonstrates the effect of varying β values during training, while the right figure highlights the impact of search iterations in MCTS.



(a) DPO. (b) DVO.

Figure 5: Evolution of implicit rewards during DPO and DVO training. While the reward margin increases, the implicit reward of positive solutions decreases in DPO but increases in DVO.

to 85.2%, Llama-3-8B from 76.6% (Step-DPO) to 80.6%, and Llama-3-70B from 87.7% (Step-DPO) to 90.7% on GSM8K. On the more challenging MATH benchmark, these models achieve improvements of 1.6%, 2.7%, and 4.1% on the models, respectively. Notably, the performance gains are observed across different model sizes, highlighting the scalability of DVO. Moreover, all improvements are achieved without the use of external annotations, relying solely on final answer evaluations, demonstrating the efficiency and simplicity of the DVO framework.

On two out-of-domain (OOD) datasets, DVO achieved notable improvements. For Llama-3-70B-Instruct, the accuracy on AGI-Eval-Math improved from 33.8% to 38.9%. Similar gains were observed on other backbones, showcasing its robust generalization capability.

6.3 Commonsense Tasks

We further validate the effectiveness of DVO on commonsense reasoning tasks. The results are presented in Tab. 2. The experiments are conducted on Llama-3-8B-Instruct (see implementation details in App. C). DVO demonstrates strong performance and generalization capabilities in commonsense reasoning tasks, further highlighting its versatility across different reasoning paradigms.

6.4 Ablation Study

Ablation on beta β. We evaluate the impact of β on DVO performance by scanning different values on the same training set, with all models trained for a single epoch. As shown in Fig. 4a, DVO achieves the best performance when $\beta = 0.1$. Both excessively large and small values of β result in degraded performance, indicating that DVO is sensitive to the balance between KL divergence regularization and entropy constraints.

Ablation on Search Iteration. We evaluate the impact of the number of search iterations in MCTS on performance. As the number of iterations increases, the accuracy of value estimation improves. As shown in Fig. 4b, inaccurate value estimation leads to a noticeable drop in performance. Additionally, we observe that on the simpler GSM8K dataset, the model requires higher precision in value estimation. We hypothesize that this is because simpler tasks demand finer distinctions between different steps to improve reasoning performance. Overall, we find that search iterations beyond 40 generally lead to better results.

6.5 Analysis

Deep observation on implicit reward. Recent work (Chen et al., 2024c; Yuan et al., 2024; Pal et al., 2024) has shown that during DPO training, while the implicit reward margin between positive and negative responses continues to increase, the implicit reward of the positive responses gradually decreases (as shown in Fig. 5a). This phenomenon likely arises because DPO's optimization objective emphasizes maximizing the reward gap between positive and negative responses, rather than directly increasing the rewards for positive responses or decreasing those for negative ones. Consequently, DPO causes a decrease in the probability of positive data, resulting in suboptimal outcomes. We an-

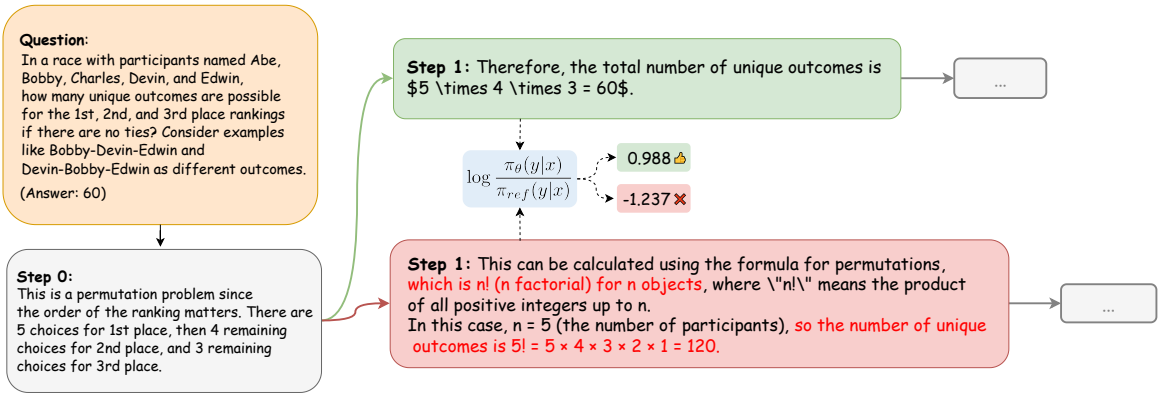


Figure 6: Step-by-step analysis of solutions generated by the reference model. The figure compares correct (top) and incorrect (bottom) CoT responses sampled from π_{ref}. Each action is evaluated based on the DVO credit defined before, with DVO credit assigned positively for correct steps and negatively for incorrect steps.

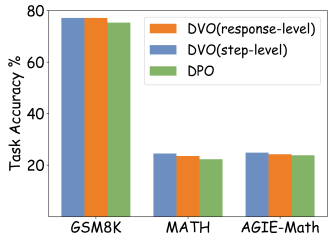


Figure 7: Different supervision granularity in DVO.

alyze the changes in implicit rewards during DVO training (as shown in Fig. 5b). Unlike DPO, we observe that the implicit rewards for the chosen responses increase steadily throughout training. This behavior is likely attributed to DVO's objective of directly aligning value functions, which effectively enhances performance on reasoning tasks by maintaining a consistent focus on improving the rewards for preferred responses.

Step level vs. response level DVO. DVO can be applied at the response level, where the entire response is treated as a single action, and optimization is performed over the entire response. This setup is similar to DPO, while the key difference is that DVO directly aligns the Q-value of the response rather than employing contrastive learning on paired data. We conduct comparative experiments on Llama-3-8B-Instruct, and the results are shown in Fig. 7. As the figure shows, response-level optimization underperforms step-level optimization, demonstrating the effectiveness of finer-grained supervision for improving reasoning. Notably, response-level optimization still outperforms DPO, suggesting that at the same granularity, optimizing for explicit value signals rather than preference relationships enables the policy model to generate high-quality responses more effectively.

Credit assignment cases. Similar as Rafailov et al. (2024), after DVO training we can take $\frac{\pi_\theta(a_t|s_t)}{\pi_{\text{ref}}(a_t|s_t)} = r(s_t, a_t) + V_\theta(s_{t+1}) - V_\theta(s_t)$ from Eq. 10 as the DVO credit, indicating the potential reward or loss from transitioning between states. We sample correct and incorrect reasoning chain from the reference model and visualize steps with the corresponding DVO credit. As illustrated in Fig. 6, each step is colored based on the DVO credit, with erroneous steps receiving low credit (darker colors). This example demonstrates that DVO has learned step-level value information, enabling it to identify faulty reasoning steps.

7 Conclusion

We introduced Direct Value Optimization (DVO), a novel framework that enhances LLMs' reasoning ability by directly optimizing reasoning paths with estimated step-wise value estimation instead of preference labels. This approach provides finer-grained supervision signals while eliminating the need for paired data construction. Practically, we generate offline samples via tree search and estimate target values using either MC estimation or a value model, then update the policy model with MSE loss. Extensive experiments on both in-domain and out-of-domain datasets demonstrate DVO's effectiveness. Notably, these improvements were achieved without requiring new query data or additional supervision. Furthermore, our analytical experiments validated the utility of step-level value information, underscoring the broader applicability of DVO across various reasoning tasks.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable feedback. This work is funded by the National Natural Science Foundation of China Key Program (Grant No. 62336006).

Limitations

The proposed DVO approach exhibits two primary limitations. First, the current implementation of DVO is confined to offline settings. However, this limitation can be addressed in future work by integrating the MCTS search process with the update step in the training pipeline, thereby enabling online operation. Second, due to computational constraints, we restrict the MCTS search width to 5 in our experiments. Theoretically, with adequate computational resources, expanding the MCTS search width would yield superior performance through two mechanisms: we will get a more accurate value estimate, while simultaneously the scale of data available for training is also larger.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2023. Llemma: An open language model for mathematics. *arXiv preprint arXiv:2310.10631*.

Guangsheng Bao, Hongbo Zhang, Cunxiang Wang, Linyi Yang, and Yue Zhang. 2025. How likely do LLMs with CoT mimic human reasoning? In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 7831–7850, Abu Dhabi, UAE. Association for Computational Linguistics.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690.

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024a. Alphamath almost zero: process supervision without process. *arXiv preprint arXiv:2405.03553*.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024b. Step-level value preference optimization for mathematical reasoning. *arXiv preprint arXiv:2406.10858*.

Huayu Chen, Guande He, Lifan Yuan, Ganqu Cui, Hang Su, and Jun Zhu. 2024c. Noise contrastive alignment of language models with explicit rewards. *Preprint*, arXiv:2402.05369.

Junying Chen, Zhenyang Cai, Ke Ji, Xidong Wang, Wanlong Liu, Rongsheng Wang, Jianye Hou, and Benyou Wang. 2024d. Huatuogpt-o1, towards medical complex reasoning with llms. *arXiv preprint arXiv:2412.18925*.

Zhihong Chen, Feng Jiang, Junying Chen, Tiannan Wang, Fei Yu, Guiming Chen, Hongbo Zhang, Juhao Liang, Chen Zhang, Zhiyi Zhang, Jianquan Li, Xiang Wan, Benyou Wang, and Haizhou Li. 2023. Phoenix: Democratizing chatgpt across languages. *arXiv preprint arXiv:2304.10453*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

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*.

Rémi Coulom. 2006. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pages 72–83. Springer.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

Xidong Feng, Ziyu Wan, Muning Wen, Ying Wen, Weinan Zhang, and Jun Wang. 2023. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. 2017. Reinforcement learning with deep energy-based policies. In *International conference on machine learning*, pages 1352–1361. PMLR.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. *ArXiv*, abs/2305.14992.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. 2024. Vineppo: Unlocking rl potential for llm reasoning through refined credit assignment. *arXiv preprint arXiv:2410.01679*.

Levente Kocsis and Csaba Szepesvári. 2006. Bandit based monte-carlo planning. In *European conference on machine learning*, pages 282–293. Springer.

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*.

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. *Preprint*, arXiv:2305.20050.

Guanlin Liu, Kaixuan Ji, Renjie Zheng, Zheng Wu, Chen Dun, Quanquan Gu, and Lin Yan. 2024. Enhancing multi-step reasoning abilities of language models through direct q-function optimization. *arXiv preprint arXiv:2410.09302*.

Hanmeng Liu, Ruoxi Ning, Zhiyang Teng, Jian Liu, Qiji Zhou, and Yue Zhang. 2023a. Evaluating the logical reasoning ability of chatgpt and gpt-4. *arXiv preprint arXiv:2304.03439*.

Jiacheng Liu, Andrew Cohen, Ramakanth Pasunuru, Yejin Choi, Hannaneh Hajishirzi, and Asli Celikyilmaz. 2023b. Making ppo even better: Value-guided monte-carlo tree search decoding. *arXiv preprint arXiv:2309.15028*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.

Arindam Mitra, Luciano Del Corro, Shweti Mahajan, Andres Codas, Clarisse Simoes, Sahaj Agarwal, Xuxi Chen, Anastasia Razdaibiedina, Erik Jones, Kriti Aggarwal, et al. 2023. Orca 2: Teaching small language models how to reason. *arXiv preprint arXiv:2311.11045*.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. 2024. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. 2024. From r to Q^*: Your Language Model is Secretly a Q-Function. *arXiv preprint arXiv:2404.12358*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Christopher D Rosin. 2011. Multi-armed bandits with episode context. *Annals of Mathematics and Artificial Intelligence*, 61(3):203–230.

John Schulman, Xi Chen, and Pieter Abbeel. 2017a. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*.

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

Amrith Setlur, Saurabh Garg, Xinyang Geng, Naman Garg, Virginia Smith, and Aviral Kumar. 2024. Rl on incorrect synthetic data scales the efficiency of llm math reasoning by eight-fold. *arXiv preprint arXiv:2406.14532*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Yu Wu, and Daya Guo. 2024. Deepseekmath: Pushing the

limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Jianhao Shen, Yichun Yin, Lin Li, Lifeng Shang, Xin Jiang, Ming Zhang, and Qun Liu. 2021. Generate & rank: A multi-task framework for math word problems. *arXiv preprint arXiv:2109.03034*.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2022. Commonsenseqa 2.0: Exposing the limits of ai through gamification. *arXiv preprint arXiv:2201.05320*.

Zhiyang Teng, Ruoxi Ning, Jian Liu, Qiji Zhou, Yue Zhang, et al. 2023. Glore: Evaluating logical reasoning of large language models. *arXiv preprint arXiv:2310.09107*.

Huaijie Wang, Shibo Hao, Hanze Dong, Shenao Zhang, Yilin Bao, Ziran Yang, and Yi Wu. 2024. Offline reinforcement learning for llm multi-step reasoning. *arXiv preprint arXiv:2412.16145*.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. *arXiv preprint arXiv:2305.04091*.

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Y Wu, and Zhifang Sui. 2023b. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *CoRR, abs/2312.08935*.

Johannes Welbl, Nelson F Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. *arXiv preprint arXiv:1707.06209*.

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

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024a. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E Gonzalez, and Bin Cui. 2024b. Buffer of thoughts: Thought-augmented reasoning with large language models. *arXiv preprint arXiv:2406.04271*.

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.

Fei Yu, Anningzhe Gao, and Benyou Wang. 2023a. Outcome-supervised verifiers for planning in mathematical reasoning. *arXiv preprint arXiv:2311.09724*.

Fei Yu, Hongbo Zhang, Prayag Tiwari, and Benyou Wang. 2023b. Natural language reasoning, a survey. *ACM Computing Surveys*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023c. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Lifan Yuan, Ganqu Cui, Hanbin Wang, Ning Ding, Xingyao Wang, Jia Deng, Boji Shan, Huimin Chen, Ruobing Xie, Yankai Lin, et al. 2024. Advancing llm reasoning generalists with preference trees. *arXiv preprint arXiv:2404.02078*.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2023. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*.

Xuan Zhang, Chao Du, Tianyu Pang, Qian Liu, Wei Gao, and Min Lin. 2024. Chain of preference optimization: Improving chain-of-thought reasoning in llms. *arXiv preprint arXiv:2406.09136*.

Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H Chi, Quoc V Le, and Denny Zhou. 2023. Take a step back: Evoking reasoning via abstraction in large language models. *arXiv preprint arXiv:2310.06117*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364*.

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

A Proof of Proposition

Proposition 1. *In general maximum entropy reinforcement learning setting, a language model parameterized by π_θ can be seen as an optimal soft Q-function under some reward.*

$$\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t) = \exp\left(\frac{1}{\beta}\left(Q^*(\mathbf{s}_t, \mathbf{a}_t) - V^*(\mathbf{s}_t)\right)\right), \quad (11)$$

Proof. The proposition is introduced in Rafailov et al. (2024), and we provide a similar proof here at the step level. Firstly, we prove that a large language model in a token-level MDP represents an optimal soft Q-function. In the token-level MDP, each action $\mathbf{a}_t$ corresponds to a token, and the action space $\mathcal{A}$ is defined as the vocabulary of the language model. At step t, the state $\mathbf{s}_t$ comprises a prompt and a sequence of generated reasoning tokens. The policy model π_θ selects the next action $\mathbf{a}_t$ based on the probability distribution $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$. To encourage robust reasoning, the reward function $r(\mathbf{s}_t, \mathbf{a}_t)$ is defined at the token level. The transition function f concatenates the current state s_t with the selected action $\mathbf{a}_t$, producing the next state $f(\mathbf{s}_t, \mathbf{a}_t) = \mathbf{s}_t \mid \mathbf{a}_t$. The discount factor γ is set to 1. We also omit explicit references in transition probabilities $P(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)$

A language model can be represented as a set of logits $\ell_\theta(\mathbf{s})$, which, combined with a temperature parameter β, define the action distribution via a softmax function:

$$\pi_\theta(\mathbf{a} \mid \mathbf{s}) = \frac{\exp(\ell_\theta(\mathbf{s})[\mathbf{a}]/\beta)}{\sum_{\mathbf{a}' \in \mathcal{A}} \exp(\ell_\theta(\mathbf{s})[\mathbf{a}']/\beta)}. \quad (12)$$

The optimal Q-function $Q^*(\mathbf{s}_t, \mathbf{a}_t)$ is defined as the expected cumulative reward when starting from state $\mathbf{s}_t$ and taking action $\mathbf{a}_t$, following the Bellman equation:

$$\begin{aligned} Q^*(\mathbf{s}, \mathbf{a}) &= r(\mathbf{s}, \mathbf{a}) + \beta \log V^*(\mathbf{s}') \\ &= r(\mathbf{s}, \mathbf{a}) + \beta \log \sum_{\mathbf{a}' \in \mathcal{A}} \exp\left(\frac{1}{\beta} Q^*(\mathbf{s}', \mathbf{a}')\right) \end{aligned} \quad (13)$$

where $s' = f(\mathbf{s}, \mathbf{a})$ is the next state. The optimal policy $\pi^*(\mathbf{a} \mid \mathbf{s})$ is derived from the optimal Q-function:

$$\pi^*(\mathbf{a} \mid \mathbf{s}) = \frac{\exp(Q^*(\mathbf{s}, \mathbf{a})/\beta)}{\sum_{\mathbf{a}' \in \mathcal{A}} \exp(Q^*(\mathbf{s}, \mathbf{a}')/\beta)}. \quad (14)$$

Given the LM policy in Eq. (12), let $Q^*(\mathbf{s}, \mathbf{a}) \triangleq \ell_\theta(\mathbf{s})[\mathbf{a}]$. Substituting $Q^*(\mathbf{s}, \mathbf{a})$ into Eq.(14), we obtain:

$$\pi_\theta(\mathbf{a} \mid \mathbf{s}) = \frac{\exp(Q^*(\mathbf{s}, \mathbf{a})/\beta)}{\sum_{\mathbf{a}' \in \mathcal{A}} \exp(Q^*(\mathbf{s}, \mathbf{a}')/\beta)}. \quad (15)$$

Thus, the LM policy $\pi_\theta(\mathbf{a} \mid \mathbf{s})$ matches the optimal policy π^* derived from $Q^*(\mathbf{s}, \mathbf{a})$ in Eq. (14). To ensure consistency with the Bellman equation, the reward function $r(\mathbf{s}, \mathbf{a})$ must satisfy:

$$r(\mathbf{s}, \mathbf{a}) = \ell_\theta(\mathbf{s})[\mathbf{a}] - \beta \log \sum_{\mathbf{a}' \in \mathcal{A}} \exp\left(\frac{1}{\beta} \ell_\theta(\mathbf{s}')[\mathbf{a}']\right), \quad (16)$$

where $\mathbf{s}' = f(\mathbf{s}, \mathbf{a})$. This ensures that the logits $\ell_\theta(\mathbf{s})$ represent the optimal Q-function. Thus the logits of the language model correspond to the optimal Q-function, and the resulting policy is optimal for this setting.

While the above argument is presented at the token level, an equivalent view is to treat multiple tokens as a single "macro-action" and define the step-level reward as the sum of the token-level rewards for those tokens. In this step-level perspective, one action $\widetilde{\mathbf{a}} = (\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_L)$ corresponds to sequentially choosing $\mathbf{a}_1, \ldots, \mathbf{a}_L$ at the token level. If we denote by π_θ the token-level optimal policy, then the probability of executing the macro-action $\widetilde{\mathbf{a}}$ under π_θ is precisely the product of its single-token probabilities: $\pi_\theta(\mathbf{a}_1 \mid \mathbf{s}) \times \cdots \times \pi_\theta(\mathbf{a}_L \mid \mathbf{s}^{(L-1)})$. Consequently, viewing several tokens as one step does not alter the overall sequence distribution or the value derived from it, the language model's logits still induce the same optimal strategy under this coarser step-level view, since the probability of any generated block of tokens and its accumulated reward remain consistent with the original token-level formulation. □

B Implementation Details

To enable LLMs to follow Step-by-step pattern, we collect self-generated data to fine-tune all backbone models. Specifically, similar to Lightman et al. (2023), we start by generating multiple solutions to 10,000 MetaMath problems in a few-shot manner and use the correct solutions to SFT the backbone models. The objective here is to avoid introducing external reasoning paths, as our focus is solely on leveraging answers to enhance the model's performance. The few-shot prompt is shown as below:

Few-shot Prompt

You are tasked with solving the provided math word problem by following these instructions:
1. Formulate and solve the equations using algebraic methods, ensuring each equation is written strictly in LaTeX syntax.
2. Document each step of your process clearly. Use double line breaks '\n\n' to separate each step and ensure that there are no double line breaks within a single step.
3. Ensure the number of reasoning steps is within 8 steps.
4. Conclude your solution by stating the final answer after the phrase 'Final Answer:'.

<|Begin Question |>
{{Question1}}
<|End Question |>

<|Begin Answer|>
{{Answer1}}
<|End Answer |>

... × n shots

<|Begin Question |>
{{QUESTION}}
<|End Question |>

<|Begin Answer|>

For each problem, we generate 64 completions and filter out responses with correct answers to train the backbone models. We then fine-tune all backbones on the self-generated data for one epoch, using a temperature of $2e{-}5$ and a batch size of 128.

For evaluation, we set the temperature to 0.7 and run three test trials in zero-shot setting, reporting the average result.

C Comparative Experiments Implementation Details

To facilitate a comprehensive comparison of the DVO with other baseline algorithms, we implement all algorithms on the same dataset. The training data are sampled from the data collected in the first round. The sampling methods vary according to the specific algorithm, detailed as follows:

- RFT: Samples positive instances from the trees based on the solution correctness, with a maximum of four positive instances per question.
- DPO: Samples paired solutions based on the solution correctness, with each question allowing for a maximum of four pairs.
- KTO: Samples both positive and negative instances based on the solution correctness, allowing for up to four positive solutions and four negative solutions per question.
- Step-DPO: Conducts step-level paired sampling of positive and negative instances based on the step value estimation, with a maximum of four pairs per question.
- DVO: Similar to KTO, it samples both positive and negative instances based on the step value estimation, with up to four positive solutions and four negative solutions per question.

All models are trained with the same set of hyperparameters. β is fixed at 0.1, λ_+/λ_- is set at 1.33 for KTO, and a learning rate of $5 \times e^{-7}$ with a cosine scheduler and 0.1 warmup ratio. Training is conducted over 3 epochs.

For commonsense reasoning, we evaluate on OBQA (Mihaylov et al., 2018), CSQA (Talmor et al., 2022), and ARC-Challenge (Clark et al., 2018) as in-domain test sets, with SciQ (Welbl et al., 2017) serving as the OOD test set. The training data is a combination of the training sets from three datasets, with approximately 7,000 examples used per round. Similar to the math reasoning tasks, we first fine-tune the backbone model on commonsense reasoning data, followed by DVO training. The data construction and training hyperparameters are identical to those used in the math reasoning tasks, with the only difference being that we performed a single round of training to validate the effectiveness of DVO.

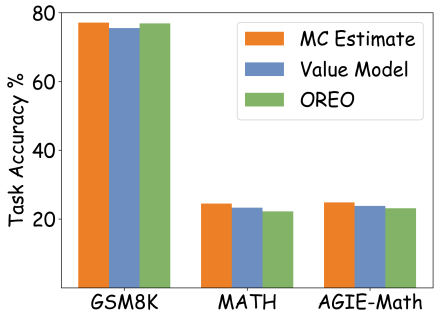


Figure 8: Compare DVO with OREO.

D Compare with OREO

Similar to DVO, the recent work OREO (Wang et al., 2024) extends its objective from maximum entropy reinforcement learning. However, unlike DVO, OREO requires joint training of a separate value model and introduces a KL regularization term. We present the results on the mathematical reasoning task in Fig. 8.