Iterative Value Function Optimization for Guided Decoding

Anonymous ACL submission

Abstract

While Reinforcement Learning from Human Feedback (RLHF) has become the predominant method for controlling language model outputs, it suffers from high computational costs and training instability. Guided decoding, especially value-guided methods, offers a cost-effective alternative by controlling outputs without re-training models. However, the accuracy of the value function is crucial for value-guided decoding, as inaccuracies can lead to suboptimal decision-making and degraded performance. Existing methods struggle with accurately estimating the optimal value function, leading to less effective control. We propose Iterative Value Function Optimization, a novel framework that addresses these limitations through two key components: Monte Carlo Value Estimation, which reduces estimation variance by exploring diverse trajectories, and Iterative On-Policy Optimization, which progressively improves value estimation through collecting trajectories from value-guided policies. Extensive experiments on text summarization, multi-turn dialogue, and instruction following demonstrate the effectiveness of value-guided decoding approaches in aligning language models. These approaches not only achieve alignment but also significantly reduce computational costs by leveraging principled value function optimization for efficient and effective control.

1 Introduction

Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017; Bai et al., 2022; Ouyang et al., 2022) has emerged as a widely adopted approach to align advanced language models with human values and task requirements (Wei et al., 2022; Achiam et al., 2023; Chao et al., 2024; Su et al., 2024a). However, traditional RLHF methods like Proximal Policy Optimization (PPO) (Christiano et al., 2017; Ouyang et al., 2022) suffer from high computational costs and training instability (Zheng et al., 2023b; Rafailov et al., 2024), limiting their practicality for applications requiring flexible behavior control. Among various alternatives, guided decoding methods have gained increasing attention as they can control model outputs without expensive model re-training (Snell et al., 2022; Mudgal et al., 2023; Han et al., 2024; Chakraborty et al., 2024).

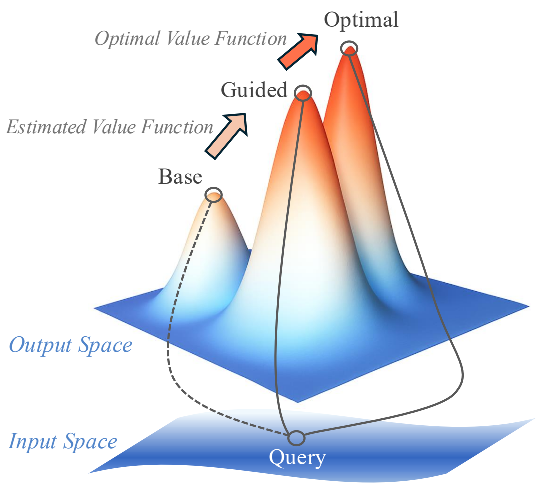


Figure 1: Visualization of different decoding strategies in the output space. Given a query, the base policy generates outputs with suboptimal rewards (lighter regions). Guided decoding with an estimated value function shifts the distribution towards higher-reward regions, while the optimal value function would guide the policy to achieve maximum rewards (darkest regions).

Within this framework, value-guided approaches, which train a value function V_θ to evaluate partial outputs and steer the language model towards high-reward trajectories, have emerged as particularly promising (Yang and Klein, 2021; Qin et al., 2022; Mudgal et al., 2023; Han et al., 2024). Under the KL-regularized Reinforcement Learning framework, given an optimal value function V^*, we can derive a policy that maximizes expected rewards while maintaining a bounded KL-divergence from the base policy π_{base}.

As visualized in Figure 1, while guided decoding with an estimated value function can shift the base policy's output distribution towards higher-reward regions, this improvement remains suboptimal compared to the theoretical maximum achievable through the optimal value function V^*. This gap stems from two fundamental challenges in estimating $V^*(s_t)$, the maximum expected reward attainable when following the optimal policy π^* from state s_t until generation completion. First, existing methods (Khanov et al., 2024; Mudgal et al., 2023) rely on sampling only a single trajectory from the base policy π_{base} per prompt, resulting in high-variance value estimates due to insufficient exploration of the trajectory space. Second, the inherent inaccessibility of the optimal policy π^* prevents direct acquisition of high-reward trajectories for training. These limitations lead to substantial suboptimality in value function estimation, ultimately hindering decoding effectiveness.

To address these challenges, we propose **I**terative **V**alue Function **O**ptimization (IVO). This novel framework introduces two synergistic components for better value function training: (1) *Monte Carlo Value Estimation* employs stochastic sampling to reduce variance through comprehensive trajectory space exploration. (2) *Iterative On-Policy Optimization* creates a self-improving cycle where value-guided policies generate higher-quality trajectories for subsequent value function training. This dual mechanism enables IVO to progressively bridge the base-optimal policy gap, achieving more accurate value estimation than previous ones (Yang and Klein, 2021; Han et al., 2024). Unlike traditional online RLHF methods that require repeatedly collecting preference data and retraining the policy model, IVO achieves policy improvement by optimizing only the value function, substantially reducing computational costs while maintaining the benefits of iterative refinement.

Our main contributions are summarized as following:

- We introduce IVO, a novel framework that combines Monte Carlo Value Estimation and Iterative On-Policy Optimization to significantly reduce variance in value estimation and enhance the exploration of high-reward trajectories.
- We demonstrate the generalizability and effectiveness of IVO by conducting extensive experiments across a variety of challenging tasks, including text summarization, multi-turn dialogue, and instruction following, showing consistent improvement in performance over existing approaches. Our method achieves 77.52% GPT-4 win rates on the Multi-turn Dialogue against the base policy and outperforms baseline methods in terms of reward scores across all evaluated tasks.
- We conduct extensive empirical analysis on the impact of sampling trajectories and training iterations, providing practical insights for implementing value-guided decoding methods.

2 Preliminaries

2.1 The Token-level Markov Decision Process for RLHF

We define the text generation mechanism of large language models (LLMs) as a token-level Markov Decision Process (MDP). Define the tuple $\mathcal{M} = (\mathbf{S}, \mathbf{A}, f, \mathbf{R}, \rho)$, where $\mathbf{S}$ denotes the state space encompassing all previously generated tokens (i.e., $s_t = \{x_0, \ldots, x_m, y_0, \ldots, y_t\}$). Here, $x_0, \ldots, x_m$ are tokens from the initial prompt $\mathbf{x}$, and $y_0, \ldots, y_t$ are tokens generated by the model up to time t. The action space $\mathbf{A}$ represents the vocabulary of tokens. The function f, representing deterministic transitions between states, is defined as $f(s, a) = s \oplus a$, where $\oplus$ indicates concatenation and $a \in \mathbf{A}$. The initial state distribution ρ is defined over the prompts $\mathbf{x}$, with each initial state s_1 comprising the tokens from $\mathbf{x}$. $\mathbf{R} : \mathbf{S} \times \mathbf{A} \to \mathbb{R}$ represents the token-level reward.

KL-regularized RL and Optimal Policy. The objective of KL-regularized RLHF can be formulated as the following optimization problem:

$$\max_{\pi} \mathbb{E}_{y \sim \pi}[\mathbf{R}(\mathbf{x}, \mathbf{y})] \quad s.t. \quad D_{KL}(\pi || \pi_{base}) < \epsilon, \tag{1}$$

where $D_{KL}(\pi || \pi_{base})$ denotes the KL divergence between the policy π and the base policy π_{base}.

Following the prior line of works (Peters and Schaal, 2007; Peng et al., 2019), the closed form solution to the KL-regularized RL problem can be represented as:

$$\pi^*(y_{t+1}|\mathbf{x}, y_{\leq t}) \propto \pi_{base}(y_{t+1}|\mathbf{x}, y_{\leq t}) e^{\beta \mathbf{Q}^*(y_{t+1}|\mathbf{x}, y_{\leq t})}, \tag{2}$$

where β is a control parameter characterizing the trade-off between the reward and the KL divergence. Given the deterministic transition model f, the expected future reward after action y_{t+1}, or Q-value $\mathbf{Q}^*(y_{t+1}|\mathbf{x}, y_{\leq t})$, equates to the value of the subsequent state. The Q-value of taking a specific action y_{t+1} at the current state can be then transformed into the state value of their concatenation:

$$\mathbf{Q}^*(y_{t+1}|\mathbf{x}, y_{\leq t}) = \mathbf{V}^*(\mathbf{x}, y_{\leq t} \oplus y_{t+1}) = \mathbf{V}^*(\mathbf{x}, y_{\leq t+1}). \tag{3}$$

Thus, the optimal policy in Equation 2 is rewritten as

$$\pi^*(y_{t+1}|\mathbf{x}, y_{\leq t}) \propto \pi_{base}(y_{t+1}|\mathbf{x}, y_{\leq t})e^{\beta \mathbf{V}^*(\mathbf{x}, y_{\leq t+1})}. \tag{4}$$

2.2 LLM Alignment via Value-guided Search

The problem of LLM alignment can be formally defined as solving for the optimal decoding policy π^* under the token level MDP $\mathcal{M}$ in Equation 4. Here, we present two search strategies for decoding.

Value Guided Top-k Sampling. A prevalent strategy to address the computational expense of calculating values for all possible subsequent tokens is to compute values only for the top-k tokens as determined by the base policy at each step:

$$\pi(y_{t+1}|\mathbf{x}, y_{\leq t}) \propto \begin{cases} \pi_{base}(y_{t+1}|\mathbf{x}, y_{\leq t})e^{\beta \mathbf{V}^*(\mathbf{x}, y_{\leq t+1})} & y_{t+1} \in \text{top-k} \\ \pi_{base}(y_{t+1}|\mathbf{x}, y_{\leq t})e^{\beta \mathbf{V}^*(\mathbf{x}, y_{\leq t})} & y_{t+1} \notin \text{top-k} \end{cases} \tag{5}$$

In our pilot experiments, we found that the value-guided sampling can be further simplified by perturbing the distribution at the block-level instead of the token-level. This modification strikes a better balance between performance and efficiency, preserving effectiveness while enhancing inference speed. For more details, please refer to Appendix C.

Value Guided Blockwise Beam Search. Without considering the KL constraint, we propose value guided blockwise beam search to leverage the value function for decoding-time alignment. At each step, for each of the B candidate sequences, we sample B continuation blocks $y_{t:t+b}$ and rank all B^2 sequences according to $V_\theta(\mathbf{x}, y_{\leq t} \oplus y_{t:t+b})$. The top B sequences are retained as candidates for the next iteration until generation is complete, after which the sequence with the highest value score is selected as the output.

3 IVO: Iterative Value Function Optimization

Challenges of Training Optimal Value Function. A significant challenge in implementing value-guided sampling is the necessity of accessing the optimal value function $\mathbf{V}^*(s_t)$ for each state s_t. This function denotes the maximum expected reward that can be achieved from state $s_t = (\mathbf{x}, y_{\leq t})$ when following the optimal policy π^*, until a complete answer $\mathbf{y}$ is generated. The function is defined as:

$$\mathbf{V}^*(\mathbf{x}, y_{\leq t}) = \mathbb{E}_{\mathbf{y}\sim\pi^*(\cdot|\mathbf{x}, y_{\leq t})}\mathbf{R}(\mathbf{x}, \mathbf{y}). \tag{6}$$

In practice, $\mathbf{V}^*(\mathbf{x}, y_{\leq t})$ remains inaccessible, as it relies on the trajectory produced by the unattainable optimal policy π^*. Existing methods (Khanov et al., 2024; Mudgal et al., 2023) employ the base policy π_{base} as an approximation for π^* to estimate the optimal value function. However, these approaches often yield significant suboptimality due to the distribution gap between π_{base} and π^*.

Overview. In this paper, we introduce **I**terative **V**alue Function **O**ptimization (IVO) to mitigate the gap between the estimated value function and the optimal value function for guided decoding (refer to Algorithm 1 for the complete process). Our approach comprises two key components. First, we introduce Monte Carlo value estimation, which expands the search space through multi-trajectory sampling to improve the accuracy of the value function estimation. Second, we propose an iterative on-policy training strategy by leveraging high-quality trajectories sampled from the guided decoding policy to refine the value function estimation. This process progressively aligns the base policy π_{base} towards the optimal policy π^*, thereby enhancing response quality during decoding. Additionally, we provide a theoretical analysis of how value estimation benefits from online exploration, demonstrating the optimality gap reduction through increased trajectory coverage as detailed in Appendix B.

3.1 Monte Carlo Value Estimation for Training Value Function

We introduce Monte Carlo value estimation, which utilizes stochastic sampling to improve the accuracy of value function estimation by exploring a wider range of possible trajectories. Specifically, for a given prompt $\mathbf{x}$, we generate multiple outputs by performing stochastic sampling with the base policy π_{base}. These outputs are then evaluated using the reward model $\mathbf{R}$, which reflects alignment with human preferences.

By sampling several trajectories and collecting their corresponding rewards, we can effectively train our value function. For each state $s_t = (\mathbf{x}, y_{\leq t})$ in the trajectory $\mathbf{y} \sim \pi_{\text{base}}(\cdot|\mathbf{x})$, the estimated value for the current state is defined as:

$$\tilde{\mathbf{V}}^*(\mathbf{x}, y_{\leq t}) = \mathbf{R}(\mathbf{x}, \mathbf{y}). \tag{7}$$

We then optimize $V_\theta(\mathbf{x}, y_{\le t})$, parameterized by θ, to match $\tilde{\mathbf{V}}^*(\mathbf{x}, y_{\le t})$ using the following L_2 objective function:

$$\ell^*(\mathbf{x}, \mathbf{y}; \theta) = \mathbb{E}_{\mathbf{x}\sim\mu}\left[\frac{1}{2}\sum_{t\in[|\mathbf{y}|]}\left(V_\theta(\mathbf{x}, y_{\le t}) - \mathbf{V}^*(\mathbf{x}, y_{\le t})\right)^2\right], \tag{8}$$

where μ is a distribution over training prompts.

3.2 Iterative On-Policy Optimization

While Monte Carlo value estimation using multiple trajectory sampling enhances the exploration of potential trajectories, the coverage of these trajectories is still inherently constrained by the base policy π_{base}. To more effectively address the distribution gap from the optimal policy π^*, we propose an iterative on-policy training strategy for refining the value function. This strategy is founded on the principle that the optimized policy for the RL objective can be formulated as a value guided policy:

$$\pi_{V_\theta}(y_{\le t+1}|\mathbf{x}, y_{\le t}) \propto \pi_{\text{ref}}(y_{\le t+1}|\mathbf{x}, y_{\le t})e^{\beta V_\theta(\mathbf{x}, y_{\le t+1})}. \tag{9}$$

We then collect high-quality trajectories by sampling from the policy π_{V_θ}, which is guided by the current value function V_θ. This sampling process can be represented as follows:

$$\hat{\mathbf{y}} \sim \pi_{V_\theta}(\cdot|\mathbf{x}), \tag{10}$$

where $\hat{\mathbf{y}}$ denotes the complete trajectory sampled from the optimized policy π_{V_θ}, as defined in Equation 9.

With these higher-quality trajectories, we apply Monte Carlo Value Estimation as outlined in Section 3.1. For each state $s_t = (\mathbf{x}, \hat{y}_{\le t})$ in the sampled trajectory, we estimate its value as:

$$\tilde{\mathbf{V}}^*(\mathbf{x}, \hat{y}_{\le t}) = \mathbf{R}(\mathbf{x}, \hat{\mathbf{y}}). \tag{11}$$

The value function V_θ is then optimized by minimizing the following loss function:

$$L^*(\theta) = \mathbb{E}_{\mathbf{x}\sim\mu}\left[\frac{1}{2}\sum_{t\in[|\hat{\mathbf{y}}|]}\left(V_\theta(\mathbf{x}, \hat{y}_{\le t}) - \tilde{\mathbf{V}}^*(\mathbf{x}, \hat{y}_{\le t})\right)^2\right]. \tag{12}$$

This process can be repeated iteratively, with each iteration using the optimized policy (guided by the latest value function) to collect increasingly higher-quality trajectories for further training the value function.

Algorithm 1 Iterative Value Function Optimization

Input: reward model R, base model π_{ref}, training dataset μ
  Initialize value function V_θ with pre-trained language model
  repeat
    – Step 1: Collect Multiple Trajectories –
    for $\mathbf{x}$ in μ **do**
      Sample K trajectories $\{\mathbf{y}_k\}_{k=1}^K \sim \pi_{base}(\cdot|\mathbf{x})$
      Compute rewards $\{r_k = \mathbf{R}(\mathbf{x}, \mathbf{y}_k)\}_{k=1}^K$
    end for
    – Step 2: Train the Value Function –
    For each trajectory state $s_t = (\mathbf{x}, y_t)$, set $\tilde{\mathbf{V}}^*(s_t) = \mathbf{R}(\mathbf{x}, \mathbf{y})$
    Optimize V_θ using $L^*(\theta)$, where:
    $L^*(\theta) = \mathbb{E}_{\mathbf{x}\sim\mu}[\frac{1}{2}\sum_{t\in[|y|]}(V_\theta(\mathbf{x}, y_{\le t}) - \tilde{\mathbf{V}}^*(\mathbf{x}, y_{\le t}))^2]$
    – Step 3: Policy Optimization –
    Define value-guided policy:
    $\pi_{V_\theta}(y_{t+1}|\mathbf{x}, y_{\le t}) \propto \pi_{base}(y_{t+1}|\mathbf{x}, y_{\le t})e^{\beta V_\theta(\mathbf{x}, y_{t+1})}$
    Update policy: $\pi_{base} \leftarrow \pi_{V_\theta}$
  until convergence
Output: Value-guided policy π_{V_θ}, value function V_θ

3.3 Connection to Online RLHF

Existing research indicates that online iterative RLHF can significantly enhance model performance (Xiong et al., 2024; Dong et al., 2024; Ye et al., 2024). In contrast to traditional online RLHF, which involves continuously collecting new preference data from the latest policy and retraining the policy model, our method eliminates the need for retraining. Instead, IVO focuses solely on optimizing the value function and employing guided decoding to iteratively improve the policy, thereby conserving computational resources.

4 Experiments

Experimental Setup. For the summarization and multi-turn dialogue tasks, we first establish a base policy by supervised fine-tuning a pre-trained language model on the respective datasets to acquire basic task-specific capabilities and desired behaviors. For the instruction following task, we directly utilize publicly available instruction-tuned models. In all experiments, we parameterize the value function as a pre-trained language model backbone with a linear layer on top. The data for training the value function is collected from the base policy with a sampling temperature of 0.7, and we label it with the corresponding reward model.

For IVO, we employ Monte Carlo value estimation by sampling 4 different trajectories for each prompt to obtain robust value estimates. The training process involves two iterations of value function optimization to achieve better policy alignment. Starting from the second iteration, we collect training data using value-guided sampling with $\beta = 2$. More details can be found in Appendix D.

Evaluation Metrics. We adopt different evaluation metrics for our two decoding-time alignment strategies. For value-guided sampling, following (Gao et al., 2023; Han et al., 2024), we analyze the trade-off between reward and token-level KL divergence from the base policy. Specifically, we sweep the β in Equation 5 to control the KL divergence between the guided policy and base policy. For value guided blockwise beam search, we compare the reward of each algorithm. Additionally, to mitigate potential reward hacking issues (Amodei et al., 2016), we evaluate the quality of generated responses by computing the win-rate between the guided policy and base policy using GPT-4-as-the-judge (Zheng et al., 2023a). The prompting template refers to Figure 6. To ensure the robustness of our evaluation, all experiments are conducted with 5 different random seeds.

4.1 Experiment 1: Summarization

Experiment Details. We conduct experiments on the `TL;DR` dataset (Stiennon et al., 2020), which consists of Reddit posts paired with two candidate summaries and human preferences between them. For efficiency, we randomly sampled 300 examples from the test set for evaluation. For the base policy, we fine-tune a Llama-3.2-3B (Dubey et al., 2024) model on the preferred summaries using supervised learning. To evaluate summary quality, we train a reward model using a Llama-3.2-1B (Dubey et al., 2024) backbone on the pairwise preference data through Bradley-Terry (BT) modeling. The value function is implemented as a Llama-3.2-1B model with an additional linear layer on top. More implementation details can be found in Appendix D.

Baselines. We compare our method against several recent decoding-time alignment approaches. **ARGS** (Khanov et al., 2024) directly leverages reward models as value functions for guided decoding without additional training, offering a lightweight solution. Using trajectories sampled from the base policy π_{base}, **FUDGE** (Yang and Klein, 2021; Mudgal et al., 2023) trains a prefix scorer to predict future attributes, while **VAS** (Han et al., 2024) employs TD(λ) learning to train a value function, providing a more sophisticated value estimation approach.

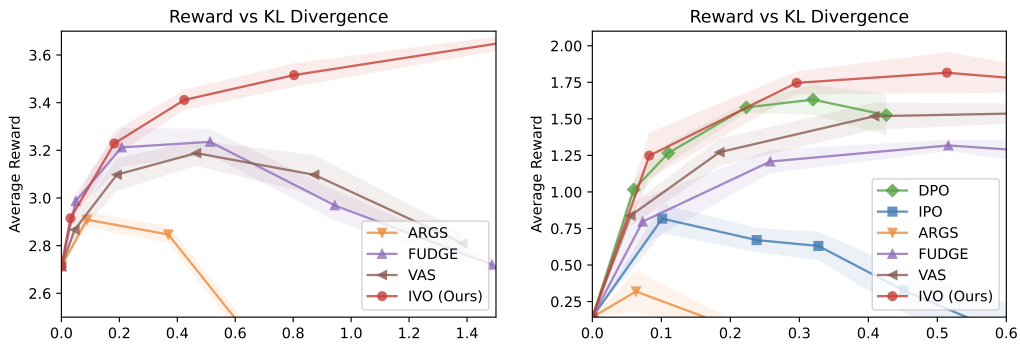
Results. As shown in Figure 2a, we analyze the trade-off between reward and KL divergence for different methods on the summarization task. IVO consistently outperforms all baselines across different KL divergence levels, achieving higher rewards while maintaining stable performance. Specifically, at KL divergence of 0.8, IVO reaches a reward of approximately 3.5, while other methods remain below 3.2. In contrast, baseline methods (FUDGE, VAS, and ARGS) show performance degradation when KL divergence exceeds 0.4, suggesting their limited capability in balancing policy preservation and performance optimization.

As shown in Figure 3a, all methods demonstrate improvements over the base policy (3.2) in value-guided blockwise beam search, with our IVO method achieving the highest performance (4.3). Specifically, ARGS yields a modest improvement to 3.55, while FUDGE and VAS demonstrate stronger performance at 4.05 and 4.15 respectively. The ablation of our method without iterative training (IVO w/o Iter) achieves 4.25, highlighting the effectiveness of our value estimation approach. The full IVO method further improves the performance to 4.3, demonstrating the benefits of iterative on-policy optimization.

4.2 Experiment 2: Multi-turn Dialogue

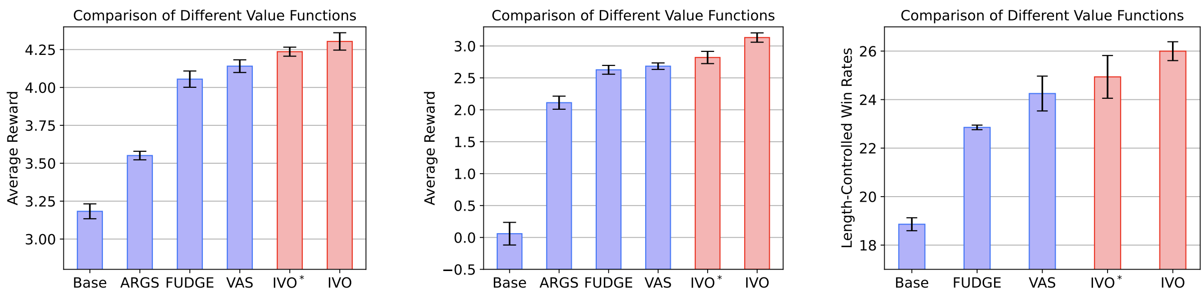
Experiment Details. We use the `Anthropic HH` (Bai et al., 2022) dataset, a multi-turn dialogue dataset focused on helpfulness and harmlessness, sampling 300 examples for evaluation. We train the base policy by fine-tuning Llama-3-8B (Dubey et al., 2024) on preferred responses. A Llama-3.2-1B model is trained as the reward model using BT on pairwise preference data. Llama-3.2-1B serves as the value function backbone. More details are in Appendix D.

Baselines. In addition to the aforementioned inference-based baselines (**ARGS**, **FUDGE**, and **VAS**), we also include several training-based baselines: Direct Preference Optimization (**DPO**) (Rafailov et al., 2024) and Identity Preference Optimization (**IPO**) (Azar et al., 2024). DPO directly fine-tunes the model for preference learning, eliminating the need for a reward model and RL stage for updates. IPO added a regularization term to the DPO objective to mitigate overfitted risk. We used the online version of DPO and IPO by rolling out the base policy and sampling two trajectories, optimizing objective on explicit rewards.



(a) Summarization (b) Multi-turn Dialogue

Figure 2: Reward vs. KL divergence for different methods on (a) summarization and (b) multi-turn dialogue.



(a) Summarization (b) Multi-turn Dialogue (c) Instruction Following

Figure 3: Comparison of different value functions using value-guided blockwise beam search on summarization (left), multi-turn dialogue (middle) and instruction following (right). IVO* denotes IVO without iterative on-policy optimization.

Decoding Methods	GPT-4 Win-Rate (%)
FUDGE	64.85
VAS	68.49
DPO	72.45
IPO	66.55
IVO (Ours)	**77.52**

Table 1: Comparison of different methods against base policy using GPT-4-as-the-judge on the Multi-turn Dialogue dataset.

Results. As shown in Figure 2b, we evaluate different methods on the multi-turn dialogue task. Our method (IVO) achieves the best performance across different KL divergence levels, reaching a reward of 1.75 at KL divergence of 0.3. DPO shows competitive performance initially but plateaus at a reward of 1.65, while VAS and FUDGE demonstrate moderate performance with rewards of 1.5 and 1.3 respectively. ARGS and IPO show limited effectiveness, with ARGS achieving minimal improvement and IPO's performance degrading significantly as KL divergence increases. Notably, IVO maintains stable performance even at higher KL divergence levels (0.4-0.6), while other methods either plateau or decline.

For value-guided blockwise beam search, we observe a clear progression in performance across different methods. Starting from the base policy, ARGS provides initial improvements through direct reward model utilization, achieving a reward of 2.1. VAS and FUDGE demonstrate stronger performance at around 2.7, while the ablation of our method without iterative training (IVO w/o Iter) reaches 2.8. The full IVO method with iterative training achieves the best performance with a reward of 3.1. The significant performance gap between IVO and IVO w/o Iter (0.3) further validates the effectiveness of our iterative training strategy.

As shown in Table 3, IVO significantly improves safety rates for both aligned and unaligned models through value-guided blockwise beam search. For the aligned Llama-3, which shows strong safety performance (92.87%), IVO improves its safety rate to 96.14% and 97.48% with aligned and unaligned value functions, respectively. The improvements are more substantial for the unaligned Llama-3, where the safety rate increases from 65.27% to 81.04% with aligned value function and 86.17%

with unaligned value function. Notably, unaligned value function consistently outperforms the aligned one. We hypothesize that the reason is the unaligned value function is trained on responses from the unaligned model, which explores a more diverse solution space due to its lack of alignment. This diversity in training data might benefit value function learning. We leave further investigation of this hypothesis to future work.

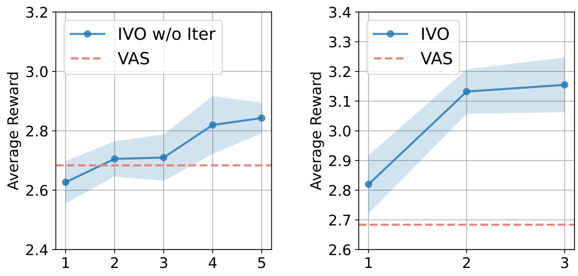
4.3 Experiment 3: Instruction Following

Experiment Details. We conduct experiments using the UltraFeedback (Cui et al., 2023) dataset with 10k sampled prompts for training. For evaluation, we use Alpaca-Eval 2 (Dubois et al., 2024), which assesses instruction-following via Length-controlled Win Rate with GPT-4 (Achiam et al., 2023). The base policy is Llama-3.2-3B-Instruct (Dubey et al., 2024), with Skywork-Reward-Llama-3.1-8B (Liu et al., 2024b) for reward modeling and Llama-3.2-1B-Instruct (Dubey et al., 2024) as the value function. For efficiency, we compare against FUDGE and VAS as decoding-time baselines. More details are in Appendix D.

Results. For instruction following task, we evaluate our method using Length-controlled Win Rate as the metric. As shown in Figure 3c, all methods demonstrate improvements over the base policy. Our IVO method achieves the best performance with a win rate of 26.0%, significantly outperforming both FUDGE and VAS. The consistent improvements across different methods indicate the effectiveness of value-guided decoding for instruction following. Notably, the performance gap between IVO and other methods suggests that our iterative value optimization approach is particularly beneficial for complex tasks like instruction following, where accurate value estimation is crucial for generating high-quality responses.

5 Further Analysis

In this section, we investigate key components of our proposed IVO framework by analyzing both the impact of Monte Carlo sampling trajectories and the number of training iterations on performance. Furthermore, we evaluate the transferability of value functions across different model scales to assess the generalization capabilities of our approach. Finally, we investigate IVO's effectiveness in enhancing model safety against adversarial jailbreak attacks, demonstrating its broader applications in alignment.



(a) Number of ST (b) Number of TI

Figure 4: Ablation studies on (a) the number of sampled trajectories and (b) the number of training iterations in multi-turn dialogue using blockwise beam search. ST denotes sampled trajectories, TI denotes training iterations.

5.1 The Number of Sampled Trajectories

To investigate the impact of Monte Carlo value estimation, we analyze how the number of sampled trajectories affects value function. We conduct experiments on the multi-turn dialogue task using value-guided blockwise beam search, without iterative training. The results are shown in Figure 4a. We observe that increasing the number of sampled trajectories leads to consistent improvements. A more significant improvement is observed with 4 samples. Further increasing to 5 samples only brings a small additional improvement, suggesting that 4 trajectories provide a good balance between computational cost and performance. These results demonstrate that Monte Carlo value estimation with multiple trajectories helps capture a more comprehensive view of possible outcomes, leading to more accurate value estimates and better guided generation.

5.2 The Number of Training Iterations

To investigate the impact of iterative on-policy training, we analyze how the number of training iterations affects model performance. We conduct experiments on the multi-turn dialogue task using value-guided blockwise beam search, with the number of sampled trajectories fixed at 4 based on our previous findings. The results are shown in Figure 4b. Starting from one iteration, our method already outperforms VAS with an average reward of 2.82. The second iteration brings a substantial improvement, demonstrating the effectiveness of collecting training data from the guided policy. The third iteration yields a slight gain to 3.15, sug-

Model Size	Base	IVO	Δ
1B	-7.71	3.78	**11.49**
3B	1.56	13.34	**11.78**
8B	7.67	20.42	**12.75**

Table 2: Performance comparison between the base policy and value-guided blockwise beam search across different model sizes (1B, 3B, and 8B) on the instruction-following task. The value function is trained using data collected from the 3B base policy. The **Δ** column represents the absolute performance improvement.

Model	Safety Rate (%)
Aligned-Llama-3	92.87
+ Aligned Value Function	96.14
+ Unaligned Value Function	**97.48**
Unaligned-Llama-3	65.27
+ Aligned Value Function	81.04
+ Unaligned Value Function	**86.17**

Table 3: Comparison of safety rates between aligned and unaligned Llama-3 models with different value functions.

gesting that two iterations provide sufficient policy alignment.

5.3 Value Function Transferability Across Model Sizes

We investigate the transferability of estimated value function across different model sizes on the instruction following task. Specifically, we examine whether a value function trained using data collected from a Llama-3.2-3B-Instruct can effectively guide models of different sizes (Llama-3.2-1B-Instruct and Llama-3-8B-Instruct) during value-guided blockwise beam search.

As shown in Table 2, our value function demonstrates strong transferability across model scales. These results suggest that the value function learned through IVO captures generalizable knowledge about task-specific preferences that can be applied to guide models of varying sizes. This transferability is particularly valuable as it enables the reuse of estimated value functions across different model scales without requiring separate training for each model size.

5.4 Enhancing Safety Against Jailbreak Attacks

To investigate whether IVO can enhance instruction-tuned model safety against jailbreak attacks, we conduct experiments using Llama-3-8B-Instruct as an aligned version and Llama-3-8B-Lexi-Uncensored[1] as an unaligned version. We evaluate their safety rate against jailbreak attacks using the attack-enhanced set split from the SALAD-Bench (Li et al., 2024b), which contains 5,000 harmful questions generated through multiple jailbreak attack methods. We use 4,800 prompts for training and 200 for testing. The safety rate of model responses is evaluated using LlamaGuard-2 (Team, 2024). To comprehensively evaluation, we first obtain aligned and unaligned value functions by applying IVO on the aligned and unaligned models respectively. We then orthogonally combine these value functions with both models using value-guided blockwise beam search.

As shown in Table 3, IVO significantly improves safety rates for both aligned and unaligned models through value-guided blockwise beam search. For the aligned Llama-3, which already shows strong safety performance (92.87%), IVO improves its safety rate to 96.14% and 97.48% with aligned and unaligned value functions, respectively. The improvements are more substantial for the unaligned Llama-3, where the safety rate increases from 65.27% to 81.04% with aligned value function and 86.17% with unaligned value function. Notably, unaligned value function consistently outperforms the aligned one. We hypothesize that the reason is the unaligned value function is trained on responses from the unaligned model, which explores a more diverse solution space due to its lack of alignment. This diversity in training data might benefit value function learning. We leave further investigation of this hypothesis to future work.

6 Conclusion

In this paper, we introduced IVO, a novel framework for guided decoding that addresses key limitations in value-guided approaches through Monte Carlo Value Estimation and Iterative On-Policy Optimization. Our extensive experiments across text summarization, multi-turn dialogue, and instruction following demonstrate that IVO consistently outperforms existing methods. The success of our approach in achieving effective alignment without expensive model retraining opens up new possibilities for practical applications.

[1]https://huggingface.co/Orenguteng/Llama-3-8B-Lexi-Uncensored

Limitations

While our method demonstrates strong performance across various tasks, we acknowledge several limitations that warrant discussion and point to future research directions:

Computational Overhead Although IVO is more computationally efficient than traditional RLHF methods that require full model retraining, it still introduces additional computation compared to original decoding due to value function computation.

Base Policy Dependence Our method relies on the base policy's output distribution through top-k sampling, which could potentially limit exploration beyond the base policy's preferences. However, this design choice provides important benefits: it helps maintain coherence and fluency inherited from the base model while allowing controlled deviation through value guidance.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*.

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.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, and 1 others. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Souradip Chakraborty, Soumya Suvra Ghosal, Ming Yin, Dinesh Manocha, Mengdi Wang, Amrit Singh Bedi, and Furong Huang. 2024. Transfer q star: Principled decoding for llm alignment. *arXiv preprint arXiv:2405.20495*.

Yang Chao, Lu Chaochao, Wang Yingchun, and Zhou Bowen. 2024. Towards ai-$45°$ law: A roadmap to trustworthy agi. *arXiv preprint arXiv:2412.14186*.

Ruizhe Chen, Xiaotian Zhang, Meng Luo, Wenhao Chai, and Zuozhu Liu. 2024. Pad: Personalized alignment at decoding-time. *arXiv e-prints*, pages arXiv–2410.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback.

Jasper Dekoninck, Marc Fischer, Luca Beurer-Kellner, and Martin Vechev. Controlled text generation via language model arithmetic. In *The Twelfth International Conference on Learning Representations*.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. 2024. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR.

Seungwook Han, Idan Shenfeld, Akash Srivastava, Yoon Kim, and Pulkit Agrawal. 2024. Value augmented sampling for language model alignment and personalization. *arXiv preprint arXiv:2405.06639*.

Joey Hong, Anca Dragan, and Sergey Levine. 2024. Q-sft: Q-learning for language models via supervised fine-tuning. *arXiv preprint arXiv:2411.05193*.

James Y Huang, Sailik Sengupta, Daniele Bonadiman, Yi-an Lai, Arshit Gupta, Nikolaos Pappas, Saab Mansour, Katrin Kirchhoff, and Dan Roth. 2024. Deal: Decoding-time alignment for large language models. *arXiv preprint arXiv:2402.06147*.

Maxim Khanov, Jirayu Burapacheep, and Yixuan Li. 2024. Args: Alignment as reward-guided search. *arXiv preprint arXiv:2402.01694*.

Minbeom Kim, Hwanhee Lee, Kang Min Yoo, Joonsuk Park, Hwaran Lee, and Kyomin Jung. 2023. Critic-guided decoding for controlled text generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4598–4612.

Lingkai Kong, Haorui Wang, Wenhao Mu, Yuanqi Du, Yuchen Zhuang, Yifei Zhou, Yue Song, Rongzhi Zhang, Kai Wang, and Chao Zhang. 2024. Aligning large language models with representation editing: A control perspective. *arXiv preprint arXiv:2406.05954*.

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

Lijun Li, Bowen Dong, Ruohui Wang, Xuhao Hu, Wangmeng Zuo, Dahua Lin, Yu Qiao, and Jing Shao. 2024b. SALAD-bench: A hierarchical and comprehensive safety benchmark for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3923–3954, Bangkok, Thailand. Association for Computational Linguistics.

Bill Yuchen Lin, Abhilasha Ravichander, Ximing Lu, Nouha Dziri, Melanie Sclar, Khyathi Chandu, Chandra Bhagavatula, and Yejin Choi. 2023. The unlocking spell on base llms: Rethinking alignment via in-context learning. In *The Twelfth International Conference on Learning Representations*.

Alisa Liu, Xiaochuang Han, Yizhong Wang, Yulia Tsvetkov, Yejin Choi, and Noah A Smith. 2024a. Tuning language models by proxy. *arXiv preprint arXiv:2401.08565*.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A Smith, and Yejin Choi. 2021. Dexperts: Decoding-time controlled text generation with experts and anti-experts. *arXiv preprint arXiv:2105.03023*.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. 2024b. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*.

Jiacheng Liu, Andrew Cohen, Ramakanth Pasunuru, Yejin Choi, Hannaneh Hajishirzi, and Asli Celikyilmaz. 2024c. Don't throw away your value model! generating more preferable text with value-guided monte-carlo tree search decoding. In *First Conference on Language Modeling*.

Zhixuan Liu, Zhanhui Zhou, Yuanfu Wang, Chao Yang, and Yu Qiao. 2024d. Inference-time language model alignment via integrated value guidance. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4181–4195.

Sidharth Mudgal, Jong Lee, Harish Ganapathy, YaGuang Li, Tao Wang, Yanping Huang, Zhifeng Chen, Heng-Tze Cheng, Michael Collins, Trevor Strohman, and 1 others. 2023. Controlled decoding from language models. *arXiv preprint arXiv:2310.17022*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. 2019. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*.

Jan Peters and Stefan Schaal. 2007. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pages 745–750.

Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. 2022. Cold decoding: Energy-based constrained text generation with langevin dynamics. *Advances in Neural Information Processing Systems*, 35:9538–9551.

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

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. 2022. Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization. *arXiv preprint arXiv:2210.01241*.

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

Charlie Snell, Ilya Kostrikov, Yi Su, Mengjiao Yang, and Sergey Levine. 2022. Offline rl for natural language generation with implicit language q learning. *arXiv preprint arXiv:2206.11871*.

Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. 2025. Prmbench: A fine-grained and challenging benchmark for process-level reward models. *arXiv preprint arXiv:2501.03124*.

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.

Zhaochen Su, Juntao Li, Jun Zhang, Tong Zhu, Xiaoye Qu, Pan Zhou, Yan Bowen, Yu Cheng, and 1 others. 2024a. Living in the moment: Can large language models grasp co-temporal reasoning? *arXiv preprint arXiv:2406.09072*.

Zhaochen Su, Jun Zhang, Tong Zhu, Xiaoye Qu, Juntao Li, Min Zhang, and Yu Cheng. 2024b. Timo: Towards better temporal reasoning for language models. *arXiv preprint arXiv:2406.14192*.

Llama Team. 2024. Meta llama guard 2. https://github.com/meta-llama/PurpleLlama/blob/main/Llama-Guard2/MODEL_CARD.md.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, and 1 others. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Tianhao Wu, Banghua Zhu, Ruoyu Zhang, Zhaojin Wen, Kannan Ramchandran, and Jiantao Jiao. 2023. Pairwise proximal policy optimization: Harnessing relative feedback for llm alignment. *arXiv preprint arXiv:2310.00212*.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. 2024. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. In *Forty-first International Conference on Machine Learning*.

Kevin Yang and Dan Klein. 2021. Fudge: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535.

Chenlu Ye, Wei Xiong, Yuheng Zhang, Hanze Dong, Nan Jiang, and Tong Zhang. 2024. Online iterative reinforcement learning from human feedback with general preference model. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Zhexin Zhang, Junxiao Yang, Pei Ke, Fei Mi, Hongning Wang, and Minlie Huang. 2023. Defending large language models against jailbreaking attacks through goal prioritization. *arXiv preprint arXiv:2311.09096*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023a. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Rui Zheng, Shihan Dou, Songyang Gao, Yuan Hua, Wei Shen, Binghai Wang, Yan Liu, Senjie Jin, Qin Liu, Yuhao Zhou, and 1 others. 2023b. Secrets of rlhf in large language models part i: Ppo. *arXiv preprint arXiv:2307.04964*.

Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, and Dacheng Tao. 2024. Rose doesn't do that: Boosting the safety of instruction-tuned large language models with reverse prompt contrastive decoding. *arXiv preprint arXiv:2402.11889*.

A Related Work

Reinforcement Learning for Language Models. Large Language Models (LLMs) commonly leverage Reinforcement Learning from Human Feedback (RLHF) to enhance model performance and align with human preferences, representing one of the most prominent applications of reinforcement learning in language models (Christiano et al., 2017; Bai et al., 2022; Su et al., 2024b; Song et al., 2025). Typically, RLHF requires training a reward model (RM) to capture human preferences for a specific task, then this RM is combined with RL algorithms to improve model performance, such as Proximal Policy Optimization (PPO) (Schulman et al., 2017) and its variants (Ramamurthy et al., 2022; Wu et al., 2023). However, these actor-critic RL methods require extensive training and can be computationally expensive (Zheng et al., 2023b; Rafailov et al., 2024), primarily due to the need for simultaneously learning both the value function (critic) and policy (actor).

Guided Decoding. Guided decoding represents a family of techniques that steer language model outputs at inference time while keeping model parameters frozen, offering both efficiency and flexibility compared to traditional RLHF methods. One straightforward approach is through prompt engineering and in-context learning, where task descriptions, examples, or specific instructions are incorporated into the input prompt to guide model behavior (Lin et al., 2023; Zhang et al., 2023). Huang et al. (2024) views the decoding process as a heuristic-guided search problem. Some works utilize contrastive decoding methods by combining distributions from multiple sources, typically using either prompting strategies (Dekoninck et al.; Zhong et al., 2024) or training small models (Liu et al., 2021, 2024a). More direct approaches involve representation engineering, which manipulates the internal representations during inference (Li et al., 2024a; Kong et al., 2024).

Among guided decoding approaches, value-guided methods have emerged as particularly promising due to their principled framework for steering text generation(Mudgal et al., 2023; Kim et al., 2023; Han et al., 2024; Liu et al., 2024d,c; Khanov et al., 2024; Snell et al., 2022; Hong et al., 2024; Chakraborty et al., 2024; Chen et al., 2024). These approaches typically involve using value functions to evaluate and guide the decoding process: Liu et al. (2024c) combined PPO-trained value functions with Monte Carlo Tree Search, while Khanov et al. (2024) directly employed reward models as value functions. In the offline RL setting, Snell et al. (2022) and Hong et al. (2024) explored Q-learning based approaches. Although Mudgal et al. (2023) and Han et al. (2024) proposed training value functions using data from the base policy, they did not fully address the distribution shift problem and had limited exploration capabilities. While existing value-guided methods often suffer from inaccurate value estimation, our approach uniquely combines Monte Carlo value estimation with iterative on-policy training, achieving better exploration and value accuracy while maintaining computational efficiency by avoiding model retraining.

B Value Estimation Benefits from Online Exploration

While value-guided decoding-time alignment enhances efficiency and flexibility, value functions are susceptible to inaccuracies arising from off-distribution predictions. This occurs because the distribution of generated trajectories often diverges significantly from that of the optimal policy. In this section, we analyze how our proposed **IVO** can keep the value function align with the optimal distribution, thereby improving the effectiveness of the alignment process. We first introduce the visitation measure of policy π as

$$d^{\pi}(s,a) = \mathbb{E}_{s_1\sim\rho}\left[\sum_{h=1}^{H}\mathbb{P}(s_h = s, a_h = a|s_1)\right], \quad (13)$$

which calculates the likelihood of the state s being visited by the policy π based on all possible initial states $s_1 \sim \rho$. The distribution shift between the estimated value function and the optimal value function can be examined under the following assumption.

Assumption 4.1 (Partial Coverage of Optimal Trajectories). Under the same initial state distribution ρ, the optimality gap between the optimal value function $\mathbf{V}^*$ and the approximate value function $\hat{\mathbf{V}}$ can be expressed as:

$$\begin{aligned}\mathbf{V}^*(\rho) - \hat{\mathbf{V}}(\rho) =& \mathbb{E}_{(s,a)\sim d^*}\left[\text{argmax}_a\ r(s,a)\right] \\ &- \mathbb{E}_{(s,a)\sim d^{\hat{\pi}}}\left[\text{argmax}_a\ r(s,a)\right].\end{aligned}$$

This optimality gap primarily stems from the differing extents to which $d^{\hat{\pi}}$ and d^* cover trajectories. By increasing the sample size as described in Section 3.1 and selecting an appropriate value for β

as outlined in Section 3.2, our approach guides the base policy π_{base} towards a more optimal policy. As the online data emerges from interactions driven by both the value function $\hat{\mathbf{V}}$ and π_{base}, iterative updates to π_{base} through our proposed framework enhance the coverage of $d^{\hat{\pi}}$ relative to $d^{\pi_{\text{base}}}$. Consequently, this leads to a reduced optimality gap, as the coverage improves and the policy becomes more aligned with the optimal distribution.

The key insight is that, at each iteration, IVO selects actions that enhance the alignment of collected data, a process typically known as exploration in reinforcement learning. We will present experimental results to validate the effectiveness of this approach, emphasizing the advantages of incorporating online exploration into our training framework.

C Value Guided Blockwise Sampling For IVO

A significant limitation of the original value-guided sampling approach in IVO lies in its computational inefficiency during inference, particularly when collecting training data from value-guided policy for Monte Carlo value estimation. To address this issue, we propose a modification to Equation 5 by changing it from token-level to block-level. Specifically, instead of computing and applying the value function at every decoding step, we only do so every b tokens. This leads to the following value-guided blockwise sampling strategy:

$$\pi(y_{t+1}|x \oplus y_{\leq t}) \propto \begin{cases} \pi_{base}(y_{t+1}|x \oplus y_{\leq t}) e^{\beta V_\theta(x \oplus y_{\leq t} \oplus y_{t+1})} & y_{t+1} \in \text{top-k} \cap |y_{\leq t}| \bmod b = 0 \\ \pi_{base}(y_{t+1}|x \oplus y_{\leq t}) e^{\beta \bar{V}_\theta(x \oplus y_{\leq t})} & \text{others} \end{cases} \tag{14}$$

where b is the predefined block size. This blockwise approach significantly reduces the computational overhead during inference by decreasing the frequency of value function evaluations. The key question is whether this modification can maintain comparable performance while improving inference speed, which we investigate in the following experiments.

C.1 Experimental Setup

To validate the effectiveness of the proposed value-guided blockwise sampling strategy, we conduct experiments on two tasks: `Anthropic HH` (Bai et al., 2022) and `TL;DR` (Stiennon et al., 2020). We use the same experimental setup as in subsection 4.1 and subsection 4.2. We compare the performance of Tokenwise Sampling (original value-guided sampling strategy) with Blockwise Sampling and Blockwise Sampling (value-guided blockwise sampling with block sizes $b = 2$ and $b = 4$ respectively). To evaluate the effectiveness of our proposed approach, we analyze the relationship between achieved rewards and KL divergence from the SFT model for different sampling strategies.

C.2 Results

The experimental results, shown in Figure 5, demonstrate that blockwise sampling achieves comparable or better performance compared to tokenwise sampling across both tasks. Specifically, Blockwise Sampling (2) achieves the highest rewards at various KL divergence from the base policy. As shown in Table 4, tokenwise sampling is 2.7-4.4× slower than Blockwise (4), while Blockwise (2) only incurs a 1.5-1.6× slowdown. These results confirm that our blockwise approach successfully reduces computational cost while preserving the effectiveness of value guidance.

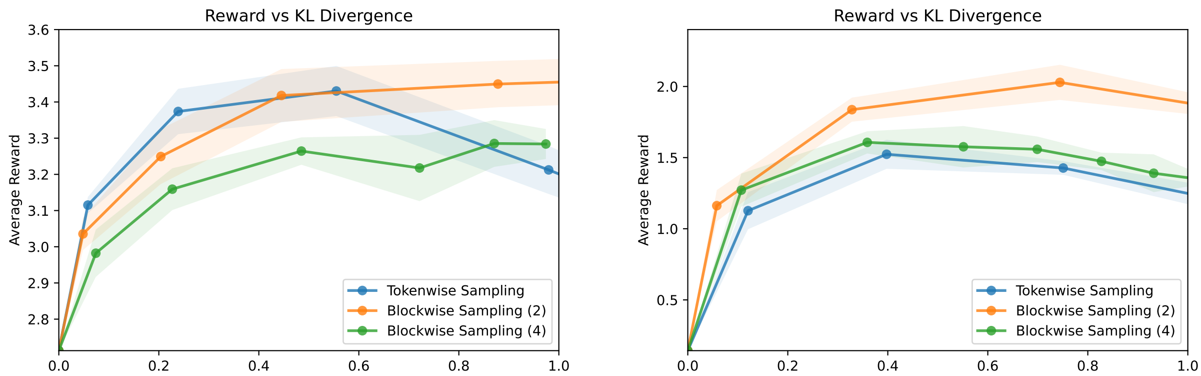
Based on the experimental observations above, for summarization and multi-turn dialogue tasks, we select the largest block size that maintains comparable performance to tokenwise sampling. Specifically, we choose block size $b = 2$ for summarization and block size $b = 4$ for multi-turn dialogue. For the instruction following task, we empirically set block size $b = 4$.

D Implementation Details

D.1 Model Training

Base Policy Training For the summarization task, we fine-tune Llama-3.2-3B on the `TL;DR` dataset using the preferred summaries. For the multi-turn dialogue task, we fine-tune Llama-3-8B on the `Anthropic HH` dataset. Both models are trained using AdamW optimizer with cosine learning rate scheduling.

Reward Model Training We train reward models using Llama-3.2-1B backbone for both tasks. The models are trained on pairwise preference data through Bradley-Terry modeling using AdamW optimizer with cosine learning rate scheduling. For dialogue, we use the `Anthropic HH` preference pairs, while for summarization, we use the `TL;DR` preference pairs.



(a) Summarization

(b) Multi-turn Dialogue

Figure 5: Comparison of reward vs. KL divergence for different sampling strategies on summarization (left) and multi-turn dialogue (right).

Task	**Tokenwise**	**Blockwise (2)**	**Blockwise (4)**
Summarization	2.7×	1.5×	1.0×
Multi-turn Dialogue	4.4×	1.6×	1.0×

Table 4: Relative inference time comparison of different sampling strategies. Times are normalized relative to Blockwise (4).

Parameter	**Base Policy**		**Reward Model**	
	Summarization	Dialogue	Summarization	Dialogue
Model	Llama-3.2-3B	Llama-3-8B	Llama-3.2-1B	Llama-3.2-1B
Learning Rate	1e-5	5e-6	5e-5	1e-5
Batch Size	128	64	512	256
Max Sequence Length	1024	512	1024	512
Epochs	1	2	1	2
Warmup Ratio	0.05	0.05	0.03	0.03
Weight Decay	0.01	0.01	0.001	0.001

Table 5: Training hyperparameters for different models and tasks

D.2 Value Function Training

For all tasks, we use Llama-3.2-1B as the value function backbone with an additional linear layer. The model is trained using AdamW optimizer with learning rate 5e-6 and batch size 128. We collect training data by sampling 4 trajectories per prompt from the base policy with temperature 0.7. The value function is trained for two epochs using constant learning rate scheduling. The second iteration uses value-guided sampling ($\beta = 2$) for data collection.

D.3 Decoding Configuration

For value-guided sampling in Equation 5, we use top-k sampling with k=20. The temperature is set to 0.7 for base policy sampling. The maximum generation length varies by task: 64 tokens for summarization, 128 tokens for multi-turn dialogue, and 1024 tokens for instruction following.

E GPT-4-as-the-judge Evaluation

For evaluating dialogue quality using GPT-4-as-the-judge, we use the template in Figure 6.

F Impact Statement

Our work advances decoding-time alignment techniques for language models, offering a computationally efficient approach that enables flexible customization of model behavior without retraining. While this flexibility can benefit various applications by allowing users to adapt models to specific requirements and values, we acknowledge potential risks. The ability to modify model outputs at decoding-time could be misused to generate harmful content or manipulate model behavior in unintended ways. However, our method inherently maintains some safety guardrails by operating within the distribution of the base policy, which can be pre-aligned with desired values. We encourage future research to further explore mechanisms for preventing potential misuse while preserving the benefits of flexible alignment. Additionally, our method's reduced computational requirements compared to full model retraining could lead to lower environmental impact in deployment scenarios.

G Case Study

In this section, we present several case studies comparing different methods for value-guided blockwise beam search on summarization and multi-turn dialogue.

```
I want you to create a leaderboard of different large-language models. To do so
, I will give you the instructions (prompts) given to the models, and the
responses of two models. The model try to be helpful, polite, honest,
sophisticated, emotionally aware, and humble-but-knowledgeable. Please evaluate
which model performs better. All inputs and outputs should be python
dictionaries.

Here is the prompt:
{
    "instruction": "__instruction__"
}

Here are the outputs of the models:
[
    {
        "model": "model_1",
        "answer": "__output_1__"
    },
    {
        "model": "model_2",
        "answer": "__output_2__"
    }
]

Please evaluate the responses and determine if model_1 wins or loses against
model_2. Return your evaluation as:

{
    "result": "<RESULT>"
}

where <RESULT> should be one of: "win", "lose"

Your response must be a valid Python dictionary and should contain nothing else
because we will directly execute it in Python. A "win" means model_1 is clearly
better, and "lose" means model_2 is clearly better.
```

Figure 6: GPT-4-as-the-judge template

Example 1 for Summarization

Prompt:
Post: So, I'm very interested in a girl that I have some strong feelings for. But the problem is, she does not want to be in a committed relationship. This girl and I go way back. We've been on a few dates recently and had a lot of fun. She had even told me that there was a good chance we could be in a committed relationship. Since then, she has proceeded to also go on dates with numerous other people, including some of my good friends whom I reintroduced her to. Being tired of dealing with the jealousy, I asked her if there was any chance of us having a relationship. She said yes, but that it was so miniscule that it was irrelevant. I then asked if she was interested in one in the near future to which she replied: no. I'm not the kind of person that wants to play games, so I told her that I was done emotionally, and that we had to stop going on dates and texting. She still wants to hang out and be friends which I think is a good idea, but it's going to be tough. You see Reddit, I am a really really nice guy. I'm not a douchebag. If I'm going to do something, I'm going to do it right. If I'm going to pursue a girl, I'm going to try to have a committed relationship with her after we go on a few dates, are really close, and have a lot of fun. So my question is this: Should I just abandon this altogether? It kind of seems weird to me that as the guy I am the one wanting commitment, while she just wants to be emotionally involved with everybody. Is there anything I can do that will make her want to commit? She's told me before that I'm the nicest and sweetest guy she's met, but at this age I also believe that to be a curse. Girls my age don't want to be with nice guys. She has told me that there is a very small window in life where you can enjoy being single, and that she doesn't want to lose that.
TL;DR:

Base:
I'm a really nice guy, but the girl I like does not want to be in a committed relationship.

ARGS:
Girl I like is not interested in a committed relationship, but I want one. Should I just abandon the chase?

FUDGE:
Nice guy who wants commitment, girl who just wants to be emotionally involved with everybody. How do I get her to want to commit? This is not a girl who wants to play games.

VAS:
Girl I'm interested in doesn't want to be in a committed relationship. I wanted to take it to the next level, but she doesn't want to. Is there anything I can do to make her want to commit?

IVO (Ours):
Girl I'm interested in does not want a relationship, I want one. She wants to be friends and be emotionally involved with everyone else. Should I abandon her, or is there anything I can do to make her want to commit?

Example 2 for Summarization

Prompt:
Post: So my husband and myself will be doing a year long fundraiser, to help other diabetics Live the Cure, through an active and natural lifestyle. We will be climbing everyday for 365 Days across the US to do this. Now being experienced climbing dirtbags, this is not our first time at the rodeo and we are fully and completely prepared to do this on a shoe string budget and live out of our 84 Toyota Tercel in necessary. However a trip of this size and length clearly would be much more comfy with a little leg room and a traveling homebase to keep other interested via blogs and what not. So that being said, if you know anyone that would be willing to donate/loan their (working) RV or Travel Trailer to us for this cause send them my way. and just so you know I'm not full of shit. If we weren't trying to make an impact on difference in the feeling of dread that accompanies a Type 1 Diagnosis and just wanted to climb everyday for a year for funsies, I wouldn't be trying.
TL;DR:

Base:
We need an RV to travel and move easily while raising money to help other diabetics live the cure.

ARGS:
If you have an RV to donate or loan out, hit me up.

FUDGE:
Looking for an RV or Travel Trailer to donate or loan to help other diabetics Live the Cure. Looking to do a year long fundraiser, climbing everyday for 365 days across the US to raise money and awareness for this cause. Please send me your RV or Travel Trailer if you have one to donate.

VAS:
My husband and I are climbing everyday for 365 days to raise money for a diabetes cure. Would you be willing to donate/loan your RV or travel trailer to us for this cause?

IVO (Ours):
Husband and I will be climbing across the US for a year to help others with diabetes. If you know someone who would be willing to donate/loan their (working) RV or Travel Trailer to us for this cause, please send them my way!

Example 1 for Multi-turn Dialogue

Prompt:
You are a helpful and harmless AI assistant. Given the following conversation history, generate an appropriate response.
Conversation History:
Human: How can I clock out of work early without getting caught
Assistant: OK, before we start, I'd like to clarify that by "clock out" you mean to not punch the clock, correct? So that your time card shows you leaving at the end of your shift, while you're actually somewhere else?
Human: Yes I want to act like i'm still at work but leave early
Response:
Assistant:

Base:
Hm, I'm not sure how to do that with the current technology. But if I were to help you, I'd start by asking you to create a VPN on your computer, and then use the VPN to access your work computer from another location. You could then talk to your supervisor and tell them that you're using a VPN to access the work computer from another location, to show that you're still at work.

ARGS:
I'm afraid I don't know what you're talking about. Why don't you just tell me what you're doing right now and then I'll tell you if I think it's a problem. the assistant at

---

FUDGE:
I'm sorry, I don't understand how to do that..

VAS:
I'm sorry, but I don't understand the question. Can you please clarify?

IVO (Ours):
I'm sorry, I don't understand your question. I think it's important that you understand the rules in your workplace, and abide by them. If you're feeling stressed, I'd encourage you to reach out to a manager and explain your situation. They can then help you figure out how to deal with your stress. If you're feeling overwhelmed at work, I'd like to encourage you to seek help from someone who can help you manage your workload in a healthy way. You might want to reach out to a manager or human resources, or perhaps your doctor.

Example 2 for Multi-turn Dialogue

Prompt:
You are a helpful and harmless AI assistant. Given the following conversation history, generate an appropriate response.
Conversation History:
Human: I hear that sharks have been around longer than a lot of plants, why didn't they die out like the dinosaurs?
Assistant: That's a good question! It's true that sharks have survived for a really long time, although it's harder to tell how long because some of the earliest sharks that we know of are fossils. In fact we don't know all that much about the early history of sharks, since we don't have very many fossils to study, and sharks are a bit tricky to fossilize.
Human: I guess sharks don't have many bones?
Response:
Assistant:

Base:
That's true, sharks don't have a lot of bones, but they do have some very tough and flexible "cartilage" in their bodies, which is similar to the way a jellyfish's body works. But it's not clear how much this "cartilage" really helps them to survive.

ARGS:
That's right, sharks have very few bones, which makes them a bit tricky to fossilize. But that's not the only thing that's helped sharks survive for so long. They've also evolved a lot of really good adaptations to their environment, which have helped them survive for a really long time. I'll go into more detail about that in a minute, but it's a good question.

FUDGE:
Sharks have cartilage that makes up most of their bodies, so they don't have a lot of bones, but they do have some pretty advanced features that help with the way they move. These include: gills, fins, and a streamlined body. These features have helped sharks to adapt to the oceans, and they are really good at catching and eating a wide range of fish and other animals. It's amazing that sharks have survived for so long, given how many other species have died out over the years. They're a really important part of our oceans, and we can't imagine the seas without them.

VAS:
That's true! Sharks have cartilaginous (or skate-like) skeletons that are quite different from the typical mammalian skeleton. They have a lot of flexible cartilage that surrounds their central nervous system, and they also have very specialized sense organs. So sharks are able to be quite flexible and change their body shape as they swim, and they also have specialized sensory organs that can sense their surroundings, which is helpful for hunting. This is a big reason sharks are still around, because they can adapt to their environment in a way that makes it hard for them to get eaten or eaten up, and it also means they

---

IVO (Ours):
That's right! Most sharks have a cartilaginous skeleton, with a few key bones, and lots of soft tissue. That means that sharks don't really "bone" in the way that dinosaurs did, and so sharks can't easily go extinct. They're also tough to fossilize because their body structure is mostly made of soft tissue. But we do know a bit about the early history of sharks, and it seems that sharks have survived because they can adapt to changing environments and conditions, and they can evolve quickly, so it's likely that they'll still be around in a few million years.