Inference-Time Language Model Alignment via Integrated Value Guidance

Zhixuan Liu[1,2*] , **Zhanhui Zhou**[1*] , **Yuanfu Wang**[1], **Chao Yang**[1†] , **Yu Qiao**[1]
[1]Shanghai Artificial Intelligence Laboratory
[2]Shanghai Jiaotong University
{liuzhixuan, zhouzhanhui, wangyuanfu, yangchao, qiaoyu}@pjlab.org.cn

Abstract

Large language models are typically fine-tuned to align with human preferences, but tuning large models is computationally intensive and complex. In this work, we introduce *Integrated Value Guidance* (IVG), a method that uses implicit and explicit value functions to guide language model decoding at token and chunk-level respectively, efficiently aligning large language models purely at inference time. This approach circumvents the complexities of direct fine-tuning and outperforms traditional methods. Empirically, we demonstrate the versatility of IVG across various tasks. In controlled sentiment generation and summarization tasks, our method significantly improves the alignment of large models using inference-time guidance from `gpt2`-based value functions. Moreover, in a more challenging instruction-following benchmark AlpacaEval 2.0, we show that both specifically tuned and off-the-shelf value functions greatly improve the length-controlled win rates of large models against `gpt-4-turbo` (e.g., $19.51\% \rightarrow 26.51\%$ for `Mistral-7B-Instruct-v0.2` and $25.58\% \rightarrow 33.75\%$ for `Mixtral-8x7B-Instruct-v0.1` with Tulu guidance).

1 Introduction

Learning-based algorithms have become the standard for aligning large language models (LLMs) with human preferences, as evidenced by numerous studies (Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022; Rafailov et al., 2024b; Azar et al., 2024). Despite their success, fine-tuning LLMs is notably resource-intensive and poses implementation challenges (Rafailov et al., 2024b). These challenges have catalyzed the development of inference-time alignment methods that maintain LLMs in a frozen state and guide their decoding during testing (Mitchell et al., 2023; Liu et al., 2024; Mudgal et al., 2023; Kim et al., 2023; Huang et al., 2024; Gao et al., 2023; Beirami et al., 2024).

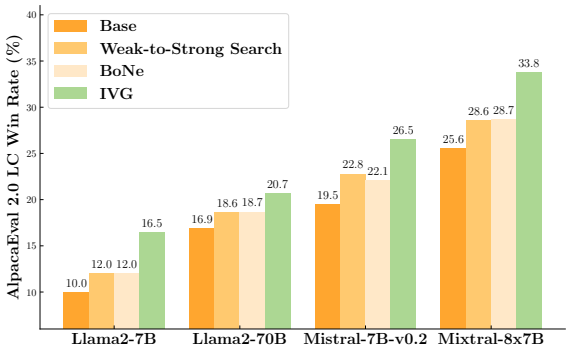

Figure 1: Illustration of Integrated Value Guidance (IVG) with parameters $W, K, L = 2, 2, 30$. Weak-to-Strong Search (Zhou et al., 2024c) denotes the results with the same parameters. BoNe denotes Best-of-N Sampling ($N = 4$) with explicit values.

Value functions, which assess the quality or alignment of generated text with desired criteria, have proven effective for inference-time alignment in two primary forms: (1) implicit value functions, represented by the log-probability differences between fine-tuned and base models (Rafailov et al., 2024a; Mitchell et al., 2023; Liu et al., 2024; Zhou et al., 2024a; Liu et al., 2021), and (2) explicit value functions, developed through direct training (Mudgal et al., 2023; Yang and Klein, 2021). Our empirical analysis reveals a significant performance discrepancy between these functions at different granularity levels of inference alignment (Section 5): explicit value functions excel at chunk-level evaluation, whereas implicit value functions are more effective at the token-level manipulation.

Recognizing this performance discrepancy, we introduce a novel algorithm called Integrated Value Guidance (IVG) to harness the strengths of both value function types. IVG combines the strengths of implicit and explicit value functions by applying implicit value functions to token-level sampling and explicit value functions to chunk-level

*Contribute equally.
†Corresponding author.

Findings of the Association for Computational Linguistics: EACL 2024, pages 4181–4195

beam search (Mudgal et al., 2023; Zhou et al., 2024c). IVG offers two significant advancements: (1) It integrates the distinctive performances of the two value function types across different granular strategies, thereby validating our theoretical models through empirical tests. (2) It introduces a robust inference-time alignment method, outperforming similar existing techniques in various evaluations. Figure 2 illustrates the IVG method.

Empirically, IVG demonstrates its versatility in tasks such as controlled-sentiment generation (Maas et al., 2011a) and summarization (Stiennon et al., 2020), where using small models like `gpt2` with 124M parameters is effective in guiding larger models from the GPT-2 series (Radford et al., 2019) to achieve competitive results. Further, in challenging instruction-following benchmarks such as *AlpacaEval 2.0* (Dubois et al., 2024), both open-source models (e.g., Tulu guidance) and our fully trained models (e.g., Ultra guidance) significantly enhance the length-controlled win rates of larger models against competitors like `gpt-4-turbo` (e.g., `Mistral-7B-Instruct-v0.2` (Jiang et al., 2023) from 19.51 to 26.51 and `Mixtral-8x7B-Instruct-v0.1` (Mistral AI team, 2023) from 25.58 to 33.75).

2 Related Work

Large unsupervised language models, trained on vast internet-scale datasets, have demonstrated remarkable capabilities (Chowdhery et al., 2023; Brown et al., 2020; Touvron et al., 2023a). Nonetheless, aligning these models with human values remains challenging. Traditionally, alignment is achieved through fine-tuning based on human evaluations of model-generated responses (Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022; Rafailov et al., 2024b; Touvron et al., 2023b; AI@Meta, 2024; Bai et al., 2022a,b). While effective, this method demands significant computational and engineering resources. Moreover, the diversity of human values complicates the creation of universally aligned models (Ouyang et al., 2022; Zhou et al., 2024b; Mudgal et al., 2023; Rame et al., 2024; Jang et al., 2023; Wang et al., 2024).

In response to these challenges, we propose an inference-time alignment approach that freezes pre-trained models while modulating their outputs through a decoding phase managed by smaller, specialized models. This strategy minimizes the need for extensive retraining and adapts more readily to individual preferences.

The conceptual framework for aligning language models at inference time is rooted in the use of value functions. Implicit value functions, as described by Rafailov et al. (2024a), proposed a token-level Markov Decision Process to adjust language model outputs based on the log-likelihood differences between fine-tuned and base models. Mudgal et al. (2023) introduced a method that leverages explicit value functions trained through a KL-regularized reinforcement learning objective, which acts as a prefix scorer. Our empirical findings, detailed in Section 5, underscore that while implicit value functions excel at refining token-level nuances, explicit value functions provide superior sequence-level contextual understanding.

These insights motivate our method, which integrates these value functions to enhance model alignment with human preferences during the decoding phase of model output generation.

3 Preliminaries

In this section, we introduce the mathematical formulation of aligning large language models (LLMs) with human preferences and then describe the implicit and explicit value functions used in our approach.

3.1 Large Language Model Alignment with Human Preferences

Aligning large language models is commonly formulated as a Kullback-Leibler (KL)-constrained optimization problem (Ziegler et al., 2019):

$$\arg\max_{\pi} \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}),\mathbf{y}\sim\pi(\mathbf{y}|\mathbf{x})}\Big[r(\mathbf{x},\mathbf{y}) - \mathbb{D}_{\mathrm{KL}}\left(\pi(\mathbf{y}\mid\mathbf{x})\|\pi_{\mathrm{ref}}(\mathbf{y}\mid\mathbf{x})\right)\Big], \tag{1}$$

where $p(\mathbf{x})$ denotes the distribution of prompts, $\mathbf{y}$ denotes the responses generated by the language model, r is the preference reward function induced from a preference datasets $\mathcal{D} = \{(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l)\}$, and $\mathbb{D}_{\mathrm{KL}}$ denotes the KL-divergence that constrains deviations from a reference model π_{ref}.

3.2 Implicit and Explicit Value Functions

The value function estimates the expected terminal reward $r(\mathbf{x}, \mathbf{y})$ when following the policy π from a given state $(\mathbf{x}, \mathbf{y}_{\leq t})$:

$$V(\mathbf{x}, \mathbf{y}_{\leq t}) = \mathbb{E}_{\mathbf{y}\sim\pi(\cdot|\mathbf{x},\mathbf{y}_{\leq t})}\left[r(\mathbf{x},\mathbf{y})\right]. \tag{2}$$

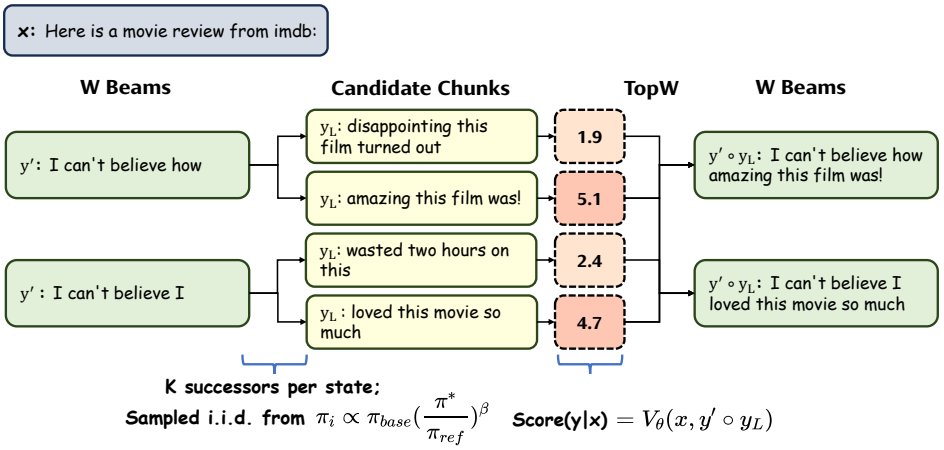

Figure 2: Illustration of Integrated Value Guidance (IVG) with beam width $W = 2$, successors per state $K = 2$, and chunk length $L = 5$.

Different forms of the value function and optimization objectives lead to various interpretations. In this work, we demonstrate both the implicit and explicit forms:

Implicit Value Function. The implicit value function is defined under the optimal policy π^* and is derived from the differences in log probabilities between the tuned model π^* and the reference model π_{ref}, as achieved by an alignment algorithm such as Direct Preference Optimization (DPO) (Rafailov et al., 2024b). Specifically, the implicit value function evaluating a partial response sequence $\mathbf{y}_{\leq t}$ is given by:

$$V^*(\mathbf{x}, \mathbf{y}_{\leq t}) - V^*(\mathbf{x}) = \log \frac{\pi^*(\mathbf{y}_{\leq t} \mid \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_{\leq t} \mid \mathbf{x})}. \quad (3)$$

Explicit Value Function. The explicit value function is a directly trained prefix scorer, which can be approximated via maximum likelihood estimation on the offline preference dataset. Specifically, the explicit value function is represented as:

$$V^*(\mathbf{x}, \mathbf{y}_{\leq t}) = V_\theta(\mathbf{x}, \mathbf{y}_{\leq t}), \quad (4)$$

where V_θ is the value function parameterized by θ.

4 Method

In this section, we introduce our proposed method, Integrated Value Guidance (IVG). First, we discuss how to utilize the value function in two distinct ways: token-wise sampling (Mudgal et al., 2023) and chunk-level beam search (Zhou et al., 2024c). Next, we explain how to train the implicit and explicit value functions on the preference dataset. Finally, we present the overall inference-time alignment process and analyze the computational efficiency of our method.

4.1 Value Function Guided Sampling and Search Strategies

Given a value function, we can guide the sampling and search strategies in two ways: token-wise sampling and chunk-level beam search.

4.1.1 Token-wise Sampling

In the token-wise sampling strategy, we use the value function to adjust the sampling distribution of the next token. Specifically, we sample the next token y_t according to the following distribution:

$$\begin{aligned}&\pi(\mathrm{y}_t \mid \mathbf{x}, \mathbf{y}_{\leq t-1}) \propto \\ &\pi_{\text{base}}(\mathrm{y}_t \mid \mathbf{x}, \mathbf{y}_{\leq t-1}) \exp(\beta(V(\mathbf{x}, \mathbf{y}_{\leq t}) - \\ &\qquad \beta(V(\mathbf{x}, \mathbf{y}_{\leq t-1}))),\end{aligned} \quad (5)$$

where $\pi_{\text{base}}(\mathrm{y}_t \mid \mathbf{x}, \mathbf{y}_{\leq t-1})$ is the base distribution of the next token, and β is a hyperparameter that controls the strength of the value function guidance.

For the implicit value function, we have:

$$\begin{aligned}&\pi_i(\mathrm{y}_t \mid \mathbf{x}, \mathbf{y}_{\leq t-1}) \propto \\ &\pi_{\text{base}}(\mathrm{y}_t \mid \mathbf{x}, \mathbf{y}_{\leq t-1}) \left(\frac{\pi^*(\mathrm{y}_t \mid \mathbf{x}, \mathbf{y}_{\leq t-1})}{\pi_{\text{ref}}(\mathrm{y}_t \mid \mathbf{x}, \mathbf{y}_{\leq t-1})} \right)^\beta.\end{aligned} \quad (6)$$

For the explicit value function, we have:

$$\begin{aligned}&\pi_e(\mathrm{y}_t \mid \mathbf{x}, \mathbf{y}_{\leq t-1}) \propto \\ &\pi_{\text{base}}(\mathrm{y}_t \mid \mathbf{x}, \mathbf{y}_{\leq t-1}) \exp\left(\beta V_\theta(\mathbf{x}, \mathbf{y}_{\leq t})\right).\end{aligned} \quad (7)$$

4.1.2 Chunk-level Beam Search

For search-based generation, the chunk-level beam search strategy is effective for value function-guided sampling. Previous research (Zhou et al., 2024c) has shown that chunk-level beam search outperforms best-of-N (BoN) sampling and requires an effective value function to rank candidate sequences. Specifically, we rank candidate sequences according to the following score:

$$r(\mathbf{y}_{\leq t}|\mathbf{x}) \propto V^*(\mathbf{x}, \mathbf{y}_{\leq t}) - V^*(\mathbf{x}), \tag{8}$$

where $r(\mathbf{y}_{\leq t}|\mathbf{x})$ is the score of the candidate sequence $\mathbf{y}_{\leq t}$, and $V^*(\mathbf{x}, \mathbf{y}_{\leq t})$ and $V^*(\mathbf{x})$ are the expected value of the candidate sequence and the prefix, respectively.

For the implicit value function, we have:

$$r_i(\mathbf{y}_{\leq t}|\mathbf{x}) = \log \frac{\pi^*(\mathbf{y}_{\leq t}|\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_{\leq t}|\mathbf{x})}. \tag{9}$$

For the explicit value function, we have:

$$r_e(\mathbf{y}_{\leq t}|\mathbf{x}) = V_\theta(\mathbf{x}, \mathbf{y}_{\leq t}). \tag{10}$$

4.2 Training the Implicit and Explicit Value Functions

There are various methods to train the implicit and explicit value functions on the preference dataset. For the implicit value function, we derive it from the difference in log probabilities between the tuned and untuned models, regardless of how the model was trained. For the explicit value function, we can employ any reinforcement learning algorithm.

In this work, we use Direct Preference Optimization (DPO) (Rafailov et al., 2024b) to train the implicit value function and FUDGE (Mudgal et al., 2023; Yang and Klein, 2021) to train the explicit value function.

For the implicit value function, we have (Rafailov et al., 2024b):

$$\ell_{\mathcal{F}}(\mathbf{x}, \mathbf{y}^w, \mathbf{y}^l) = -\log \sigma \left(\beta \log \frac{\pi_\theta(\mathbf{y}^w \mid \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}^w \mid \mathbf{x})} - \beta \log \frac{\pi_\theta(\mathbf{y}^l \mid \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}^l \mid \mathbf{x})} \right). \tag{11}$$

For the explicit value function, we have (Mudgal et al., 2023):

$$\ell_{\mathcal{F}}(\mathbf{x}, \mathbf{y}; \theta) = \frac{1}{2} \sum_{t \in [|\mathbf{y}|]} \left(V_\theta(\mathbf{x}, \mathbf{y}_{\leq t}) - r(\mathbf{x}, \mathbf{y}) \right)^2. \tag{12}$$

4.3 Integrated Value Guidance

The token-wise sampling and chunk-level beam search strategies can be combined to enhance the alignment of large language models with human preferences. Specifically, token-wise sampling adjusts the sampling distribution of the next token using the implicit value function, while chunk-level beam search ranks candidate sequences using the explicit value function.

The IVG algorithm is illustrated in Figure 2. The key insight is that by applying the implicit value function at the token level and the explicit value function at the chunk level, we effectively leverage the strengths of both. Compared to Weak-to-Strong Search (Zhou et al., 2024c), we sample tokens from a policy adjusted by the implicit value function rather than the base policy and use the explicit value function to rank candidate sequences. Empirically, we find that this combination leads to better alignment with human preferences. We demonstrate the effectiveness of IVG in the following sections.

4.4 Implementation and Complexity

We analyze the implementation efficiency and computational complexity of the IVG method. At each time step, the main components contributing to the time complexity of IVG include: (1) The base model performs a forward pass to compute the probability distribution of the next token based on the given context. (2) The implicit value functions (including π^* and π_{ref}) perform forward passes to compute the probability distributions for the next token and calculate their difference. This difference is then combined with the base model's probability distribution to obtain the final next token distribution. (3) When the current chunk reaches the length L, we compute the value of all candidate sequences using the explicit value function V_θ, and select the top-W sequences.

Consider the complexity of a single decoding step with a context of length t tokens. The complexity is:

- If the current chunk length $\neq L$:

$$T(t) = T_{\text{base}}(t) + T_{\pi^*}(t) + T_{\pi_{\text{ref}}}(t).$$

- If the current chunk length = L:

$$T(t) = T_{\text{base}}(t) + T_{\pi^*}(t) + T_{\pi_{\text{ref}}}(t) + T_{V_\theta}(t).$$

To simplify, consider the case where $L = 1$. The total time complexity becomes:

$$T(t) = T_{\text{base}}(t) + T_{\pi^*}(t) + T_{\pi_{\text{ref}}}(t) + T_{V_\theta}(t).$$

Here, $T_{\text{base}}(t)$ and other terms represent the inference time of the respective models for a sequence of length t. Initially, due to pairwise attention computations, $T(t) = O(t^2)$. However, during generation, we can cache previous computations, reducing the complexity to $O(t)$. Therefore, the per-step inference complexity is:

$$T(t) = O(t) \times (C_{\text{base}} + C_{\pi^*} + C_{\pi_{\text{ref}}} + C_{V_\theta}),$$

where C_{base}, C_{π^*}, $C_{\pi_{\text{ref}}}$, and C_{V_θ} are constants representing the computational costs of each model.

In summary, while IVG does not change the asymptotic time complexity of the generation process, it introduces additional computational overhead due to multiple forward passes and increased memory usage. We will empirically evaluate the computational complexity of different methods in the experimental section.

5 Experiments

In this section, we empirically evaluate the ability of the proposed IVG to align large language models with human preferences using only inference-time guidance from small language models. First, in **controlled-sentiment generation** (Maas et al., 2011a) and **summarization** (Stiennon et al., 2020), we tune `gpt2` to model the desired behaviors in each task to get the implicit value function and train the explicit value function based on the same base model. Then, we use the trained implicit and explicit value functions to steer larger models of various scales (Section 5.1).

Next, in a more difficult **instruction-following** benchmark, AlpacaEval 2.0 (Dubois et al., 2024), in addition to tuning small models, we reuse the off-the-shelf open-source 7B models and their untuned versions as the implicit value function and train the explicit value function by one of the best-performance sequence-wise reward models evaluated by RewardBench (Lambert et al., 2024a). We then use them to steer a series of large models.

Baselines. Considering that some existing methods could be represented as special cases of the combinations of implicit and explicit value functions for token-wise sampling and chunk-level beam search, we compare the proposed IVG and different combinations of implicit and explicit value functions with the following baselines: (1) **Base**: the base model without any value function guidance; (2) **Best-of-N Sampling (BoN)**: BoNi uses $r = \log \pi^*(\mathbf{y} \mid \mathbf{x}) - \log \pi_{\text{ref}}(\mathbf{y} \mid \mathbf{x})$ as rewards and BoNe uses $r = V_\theta(\mathbf{x}, \mathbf{y})$ as rewards to select the highest-scoring responses among the N independent response from the frozen base language model; (3) **FT**: fine-tuning the base model on the preference dataset.

Note that many existing methods can be represented by the framework: Emulator Fine-Tuning (EFT) (Mitchell et al., 2023) can be viewed as applying only the implicit value function in token-wise sampling. Weak-to-strong search can be viewed as applying only the implicit value function in chunk-level beam search. We use the EFTi, EFTe, CBSi and CBSe to represent the corresponding combination. For example, EFTi denotes applying implicit value function for token-wise sampling, and CBSe denotes applying explicit value function for chunk-level beam search.

5.1 Controlled-Sentiment Generation & Summarization

Setup. For these two tasks, we follow the synthetic setups from (Gao et al., 2023; Lightman et al., 2023; Rafailov et al., 2024b), assuming **access to a gold reward model** r_{gold}. For controlled-sentiment generation, r_{gold} encourages positive continuations of movie reviews, while for summarization, it encourages high-quality summaries of Reddit posts. We **generate synthetic preference datasets** $\mathcal{D} = \{(\mathbf{x}, \mathbf{y}^w, \mathbf{y}^l)_i\}_{i=1}^N$ from r_{gold} with $p(\mathbf{y}^1 \succ \mathbf{y}^2 \mid \mathbf{x}) = \sigma(r_{\text{gold}}(\mathbf{x}, \mathbf{y}^1) - r_{\text{gold}}(\mathbf{x}, \mathbf{y}^2))$ to mimic human feedback (Bradley and Terry, 1952).

To **obtain the implicit value function**, we optimize `gpt2` (124M parameters) using the standard DPO pipeline (Rafailov et al., 2024b): (1) we first obtain the reference model π_{ref} through supervised fine-tuning on both chosen and rejected responses from the synthetic preference dataset, then (2) we apply DPO on the synthetic preference dataset with π_{ref} as the reference policy to obtain the optimal language model π^*.

To **obtain the explicit value function**, we train a prefix scorer V_θ using the FUDGE(Mudgal et al., 2023; Yang and Klein, 2021) algorithm on the synthetic preference dataset. (1) we first train the sequence-wise reward model $r(\mathbf{x}, \mathbf{y})$ on the synthetic preference dataset, then (2) we apply the FUDGE algorithm to train the prefix scorer V_θ with

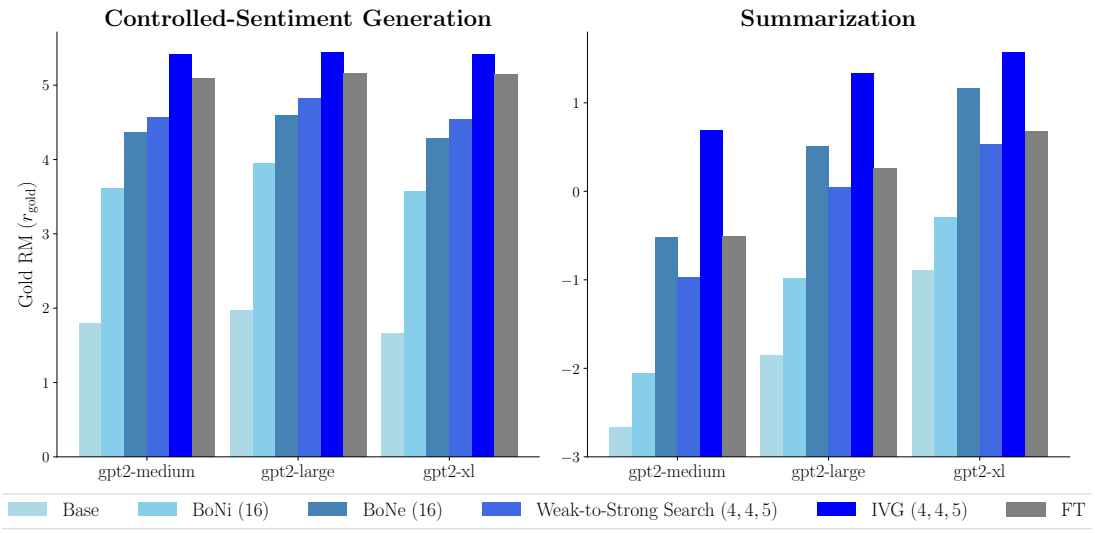

Figure 3: **The gold reward achieved for different large pre-trained models under the gpt2 guidance.** We show the mean reward across three random seeds. BoNi and BoNe denote BoN ($N = 16$) with implicit and explicit rewards, respectively; EFT (β^*) denotes the best EFT results among $\beta \in \{0.25, 0.5, 1, 2\}$; CBS denotes the results with $W, K, L = 4, 4, 5$ and implicit rewards; IVG denotes the best results with $W, K, L = 4, 4, 5$ among $\beta \in \{0.25, 0.5, 1, 2\}$.

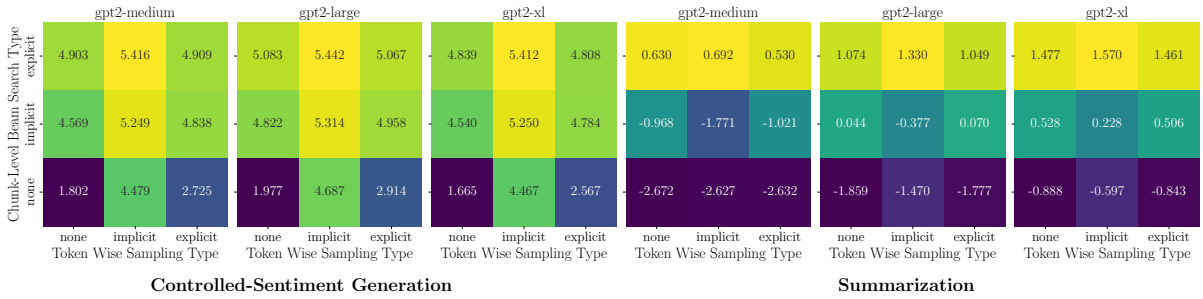

Figure 4: **The performance of different combinations of implicit and explicit value functions for token-wise sampling and chunk-level beam search in controlled-sentiment generation and summarization.** "implicit" and "explicit" denotes applying implicit and explicit value functions. "none" denotes the base model without any guidance. The number denotes the gold reward for the corresponding combination.

the sequence-wise reward model r as the reward function.

Given the implicit and explicit value functions, we use them to **steer the large pre-trained language models without additional training**. Since token-wise sampling only supports steering the model sharing the same vocabulary as the base model, here we only study on the gpt2 family models with different scales: `gpt2-mudium` (345M parameters), `gpt2-large` (774M parameters) and `gpt2-xl` (1.5B parameters). Eventually, since we have access to the gold reward model, responses can be fairly evaluated on the test split of prompts using this gold reward model.

Results. Figure 3 demonstrates IVG's outstanding performance in both controlled-sentiment generation and summarization tasks. We find that **IVG achieves the best performance among all the baselines**, showing the effectiveness of the proposed method. To assess the effectiveness of IVG, we examined how different combinations of implicit and explicit value functions perform in two tasks, as depicted in Figure 4. In chunk-level beam search, the explicit value function significantly enhances performance in both tasks, whereas the implicit value function shows lesser improvements. Conversely, in token-wise sampling, the implicit value function notably boosts performance in the controlled-sentiment generation task, but the explicit value function has a minimal impact. This distinction likely arises because the controlled-sentiment generation task primarily requires ad-

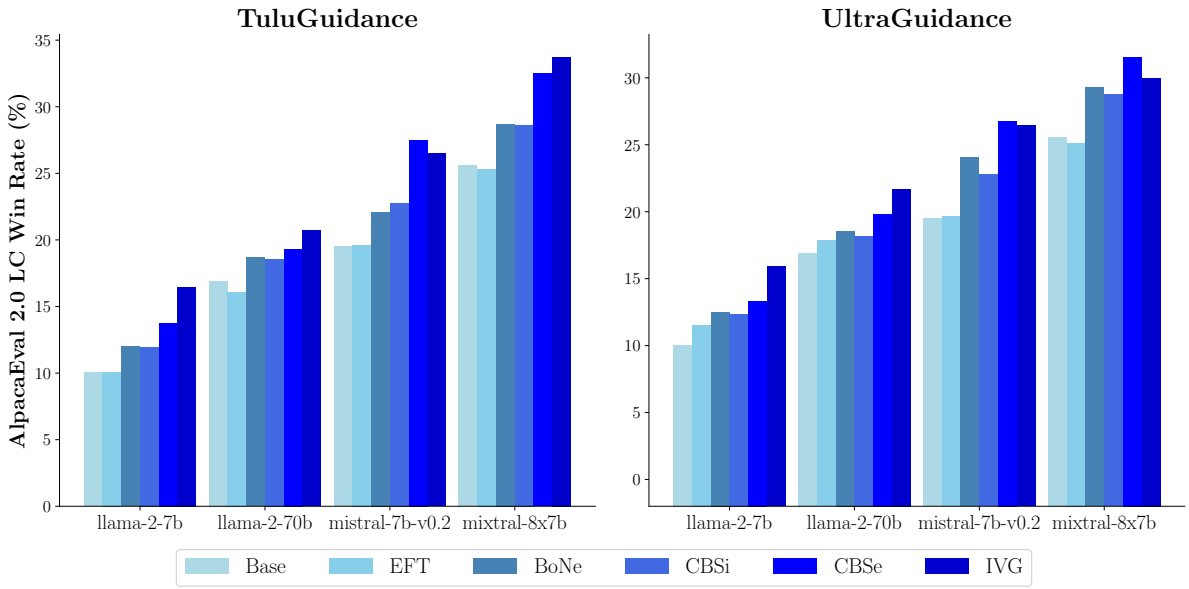

Figure 5: **The length-controlled win rates (LC Win Rates)** against gpt-4-turbo for various instruction-tuned models under TuluGuidance and UltraGuidance. BoNe denote BoN ($N = 16$) with explicit rewards, respectively; EFTi (β^*) denotes the results with $\beta_i = 1.0$ for Tulu guidance and $\beta_i = 1.5$ for Ultra guidance which are the best parameters we evaluated on `Llama-2-7b-chat-hf`. CBSi and CBSe denote the results with implicit and explicit value functions with $W, K, L = 2, 2, 30$. IVG denotes the best results with $W, K, L = 2, 2, 30$ with $\beta = \beta_i$. BoNi and EFTe are not shown due to their weak performance. More results are shown in appendix B.

justments at the token level (e.g., "dislike" $\rightarrow$ "like"), whereas summarization demands a focus on broader contextual information. The results suggest that the explicit value function excels in chunk-level beam search, while the implicit value function performs better in token-wise sampling. By integrating both value functions, IVG achieves superior performance in both tasks.

5.2 Instruction-Following

Setup. We evaluate IVG on AlpacaEval 2.0 (Dubois et al., 2024), a single-turn instruction-following benchmark comprising 805 prompts from various open-source datasets. Here, unlike steering *pre-trained* models (e.g., `Llama-2-7b`), we utilize *instruction-tuned* models (e.g., `Llama-2-7b-chat`) due to their need for additional alignment as per (Ji et al., 2024).

For smaller models, we adopt two strategies to derive implicit and explicit value functions: (1) **Tulu guidance**: Utilizing `tulu-2-dpo-7b` and its baseline `tulu-2-7b` for the implicit function, and training `FsfairX-LLaMA3-RM-v0.1` on the `UltraFeedback` dataset for the explicit function. (2) **Ultra guidance**: Fine-tuning `Llama-2-7b` via Direct Preference Optimization (DPO) (Rafailov et al., 2024b) on `UltraFeedback` for both implicit and explicit functions. All the models use the Llama-2 tokenizer.

Target instruction-tuned models include `Llama-2-7b-chat-hf`, `Llama-2-70b-chat-hf`, `Mistral-7B-Instruct-v0.2` and `Mixtral-8x7B-Instruct-v0.1`. To manage computational costs, we refrained from directly fine-tuning large models. We explored various valid combinations of implicit and explicit value functions, such as CBSe for chunk-level beam search with explicit value function. Language model responses are evaluated by their length-controlled win rates (LC WR) against `gpt-4-turbo`, with `gpt-4-turbo` serving as the judge.

Results. Figure 5 demonstrates that IVG consistently performs well. Notably, applying an explicit value function to chunk-level beam search significantly enhances outcomes, whereas an implicit value function improves results when applied to token-wise sampling, albeit less effectively for larger models such as `Mixtral-8x7B-Instruct-v0.1`. These findings confirm the theoretical trade-offs between explicit and implicit value functions.

Ablation. We present an ablation study evaluating the inference speed of various methods applied to instruction-following tasks using a single sample. We used TuluGuidance to guide the

`Llama-2-70B-chat-hf`, and the results are illustrated in Figure 6. Our empirical findings corroborate the theoretical analysis. The additional time overhead associated with IVG is primarily due to the extra inference steps required by both the implicit and explicit reward models. Notably, the implicit reward model incurs significantly higher inference costs relative to the explicit model, which can be attributed to the markedly higher forward pass frequency inherent to the implicit approach. The Chunk-level Beam Search (CBS) with hyperparameters $W = 2$, $K = 2$, and $L = 30$ demonstrates a substantial efficiency advantage over Emulator Fine-Tuning (EFT). CBS achieves superior optimization results while incurring only few additional time costs. In contrast, EFT, despite its higher time overhead, yields only modest improvements across certain models. Therefore, in practice, employing Chunk-Level Beam Search alone may represents a more efficient choice.

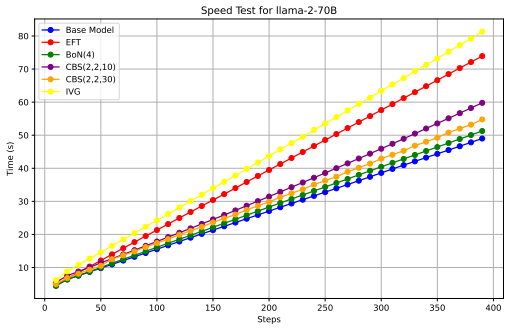

Figure 6: Inference Speed Comparison for `Llama-2-70B-chat-hf`. The figure illustrates the inference time across various methods over a range of steps.

6 Discussion

We have presented Integrated Value Guidance (IVG), a method that combines implicit and explicit value functions, applied to token-wise sampling and chunk-level beam search. We conducted experiments on synthetic tasks and instruction-following task and found that IVG achieved the best performance in both tasks. We also explored the performance of different combinations of implicit and explicit value functions in the two tasks and found that the explicit value function applied to chunk-level beam search can significantly improve the results, while the implicit value function applied to token-wise sampling can improve the results. IVG combines the advantages of both, so it achieves the best performance in both tasks.

Limitations. Our work primarily focuses on enhancing the alignment capabilities of large language models through the integration of implicit and explicit value functions. This approach introduces several complex questions that extend beyond the current scope of our research:

1. Our methodologies have been limited to using the DPO (Rafailov et al., 2024b) and FUDGE (Mudgal et al., 2023; Yang and Klein, 2021) algorithms for training and deriving the implicit and explicit value functions. It remains unclear whether incorporating other large model alignment strategies or offline reinforcement learning algorithms for token-wise sampling and chunk-level beam search might influence our findings. This aspect warrants additional experimental investigation.

2. Although we have detailed the empirical outcomes associated with the Implicit and Explicit Value Functions, a theoretical framework that explicates these results is conspicuously absent. Developing a theoretical understanding to underpin these empirical findings is an essential next step for further research.

7 Acknowledgement

This work is supported in part by the National Key R&D Program of China (NO.2022ZD0160102).

References

AI@Meta. 2024. Llama 3 model card.

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, et al. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022b. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Ahmad Beirami, Alekh Agarwal, Jonathan Berant, Alexander D'Amour, Jacob Eisenstein, Chirag Nagpal, and Ananda Theertha Suresh. 2024. Theoretical guarantees on the best-of-n alignment policy. *arXiv preprint arXiv:2401.01879*.

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.

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

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

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. *arXiv preprint arXiv:2310.01377*.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. 2023. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*.

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.

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

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A Smith, Iz Beltagy, et al. 2023. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*.

Joel Jang, Seungone Kim, Bill Yuchen Lin, Yizhong Wang, Jack Hessel, Luke Zettlemoyer, Hannaneh Hajishirzi, Yejin Choi, and Prithviraj Ammanabrolu. 2023. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*.

Jiaming Ji, Boyuan Chen, Hantao Lou, Donghai Hong, Borong Zhang, Xuehai Pan, Juntao Dai, and Yaodong Yang. 2024. Aligner: Achieving efficient alignment through weak-to-strong correction. *arXiv preprint arXiv:2402.02416*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

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.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. 2024a. Rewardbench: Evaluating reward models for language modeling. https://huggingface.co/spaces/allenai/reward-bench.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. 2024b. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*.

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

Alisa Liu, Xiaochuang Han, Yizhong Wang, Yulia Tsvetkov, Yejin Choi, and Noah A Smith. 2024.

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*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011a. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011b. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Mistral AI team. 2023. Mixtral of experts. https://mistral.ai/news/mixtral-of-experts/. Accessed: June 10, 2024.

Eric Mitchell, Rafael Rafailov, Archit Sharma, Chelsea Finn, and Christopher D Manning. 2023. An emulator for fine-tuning large language models using small language models. *arXiv preprint arXiv:2310.12962*.

Sidharth Mudgal, Jong Lee, Harish Ganapathy, YaGuang Li, Tao Wang, Yanping Huang, Zhifeng Chen, Heng-Tze Cheng, Michael Collins, Trevor Strohman, et al. 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, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners.

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

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

Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. 2024. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. *Advances in Neural Information Processing Systems*, 36.

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.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Haoxiang Wang, Yong Lin, Wei Xiong, Rui Yang, Shizhe Diao, Shuang Qiu, Han Zhao, and Tong Zhang. 2024. Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. *arXiv preprint arXiv:2402.18571*.

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. *Preprint*, arXiv:2312.11456.

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, Online. Association for Computational Linguistics.

Zhanhui Zhou, Jie Liu, Zhichen Dong, Jiaheng Liu, Chao Yang, Wanli Ouyang, and Yu Qiao. 2024a. Emulated disalignment: Safety alignment for large language models may backfire! *arXiv preprint arXiv:2402.12343*.

Zhanhui Zhou, Jie Liu, Jing Shao, Xiangyu Yue, Chao Yang, Wanli Ouyang, and Yu Qiao. 2024b. Beyond one-preference-fits-all alignment: Multi-objective direct preference optimization. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 10586–10613.

Zhanhui Zhou, Zhixuan Liu, Jie Liu, Zhichen Dong, Chao Yang, and Yu Qiao. 2024c. Weak-to-strong

search: Align large language models via searching over small language models. *arXiv preprint arXiv:2405.19262*.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. 2023. Starling-7b: Improving llm helpfulness & harmlessness with rlaif.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

A Experimental Setup Details

We adopt the experimental setup from Zhou et al. (2024c).

A.1 Controlled-Sentiment Generation & Summarization

A.1.1 Model Specification

The models are specified in the following table:

Models and Links
`gpt2` (124M) (Radford et al., 2019) `https://huggingface.co/openai-community/gpt2`
`gpt2-medium` (345M) (Radford et al., 2019) `https://huggingface.co/openai-community/gpt2-medium`
`gpt2-large` (774M) (Radford et al., 2019) `https://huggingface.co/openai-community/gpt2-large`
`gpt2-xl` (1.5B) (Radford et al., 2019) `https://huggingface.co/openai-community/gpt2-xl`

Table 1: Models and their links

A.1.2 Hyperparameters Specification

In our approach, we use fixed hyperparameters across all tested models to ensure consistency. During decoding, we set the temperature to $T = 0.7$, with top-k = None and top-p = 1.0. For chunk-level beam search, the parameters are configured as follows: beam width $W = 4$, successors per state $K = 4$, and chunk length $L = 5$. To maintain computational fairness, we set the number of samples N to 16 for the BoN sampling. For EFT, we report the best results obtained from $\beta \in \{0.25, 0.5, 1, 2\}$.

A.1.3 Compute Resources

Evaluation occurs over 1000 test prompts using a single NVIDIA A100 GPU.

A.1.4 Gold Reward Models

We follow a synthetic setup where gold reward models simulate human evaluations by generating binary preference labels (Gao et al., 2023; Lightman et al., 2023; Rafailov et al., 2024b).

For controlled-sentiment generation, we utilize the publicly accessible `distilbert-imdb` as our gold reward model r_{gold}. The `distilbert-imdb` is a fine-tuned classifier p on the `imdb` dataset (Maas et al., 2011b), designed to assess the sentiment of movie reviews. We define the gold reward r_{gold} as $\log p(\text{positive} \mid x, y) - \log p(\text{negative} \mid x, y)$, promoting positive sentiment reviews. Synthetic preferences are collected using the truncated movie reviews as prompts $\mathbf{x}$, and pairwise completions from `gpt2-imdb`, ranked by $p(\mathbf{y}_1 \succ \mathbf{y}_2 \mid \mathbf{x}) = \sigma(r_{\text{gold}}(\mathbf{x}, \mathbf{y}_1) - r_{\text{gold}}(\mathbf{x}, \mathbf{y}_2))$.

For the summarization task, we fit a reward model on the `summarize_from_feedback` dataset (Stiennon et al., 2020) as our gold reward model r_{gold}. This model is specifically fine-tuned from `Llama-2-7b` with a linear projection head and binary cross-entropy loss. The training parameters include a batch size of 32, a learning rate of `1e-5` for the projection head, and `5e-6` for the other parameters, conducted over one epoch with a cosine learning rate schedule. Synthetic preferences are generated by relabeling pairwise responses in the original dataset, using $p(\mathbf{y}_1 \succ \mathbf{y}_2 \mid \mathbf{x}) = \sigma(r_{\text{gold}}(\mathbf{x}, \mathbf{y}_1) - r_{\text{gold}}(\mathbf{x}, \mathbf{y}_2))$.

Both gold reward models exhibit high validation accuracies of 0.928 and 0.736, respectively, indicating a strong alignment with human judgment.

A.1.5 Direct Tuning Details

Direct tuning on the synthetic preferences $\mathcal{D} = \{(\mathbf{x}, \mathbf{y}^w, \mathbf{y}^l)_i\}_{i=1}^N$ involves two stages: Supervised Fine-Tuning (`SFT`) and Direct Preference Optimization (`DPO`) (Rafailov et al., 2024b). During `SFT`, models are trained on both selected and rejected responses using a batch size of 64, a learning rate of `2e-5`, and a cosine learning rate schedule over one epoch. During `DPO`, we use a $\beta = 0.1$, batch size of 256, a learning rate of `1e-6`, and a cosine learning rate schedule over one epoch.

A.1.6 Prompt Template for Sampling from Base Models

For sentiment-controlled generation, we use a zero-shot prompt:

```
Here is a movie review from imdb: {prompt}
```

For summarization, we use a two-shot prompt (the exemplars are selected arbitrarily):

```
{examplar[1].prompt}TL;DR: {examplar[1].
response}
{examplar[2].prompt}TL;DR: {examplar[2].
response}
   {prompt}TL;DR:
```

A.2 Instruction Following

A.2.1 Model Specification

The following table lists the models and their corresponding links.

Models and Links
`tulu-2-dpo-7b` (Ivison et al., 2023) https://huggingface.co/allenai/tulu-2-dpo-7b
`tulu-2-7b` (Ivison et al., 2023) https://huggingface.co/allenai/tulu-2-7b
`Llama-2-7b-chat` (Touvron et al., 2023b) https://huggingface.co/meta-llama/Llama-2-7b-chat-hf
`Llama-2-70b-chat` (Touvron et al., 2023b) https://huggingface.co/meta-llama/Llama-2-70b-chat-hf
`Mistral-7B-Instruct-v0.2` (Jiang et al., 2023) https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2
`Mixtral-8x7B-Instruct-v0.1` (Mistral AI team, 2023) https://huggingface.co/mistralai/Mixtral-8x7B-Instruct-v0.1
`FsfairX-LLaMA3-RM-v0.1` (Dong et al., 2023; Xiong et al., 2024) https://huggingface.co/sfairXC/FsfairX-LLaMA3-RM-v0.1

Table 2: Models and their links

A.2.2 Hyperparameters Specification.

We use fixed hyperparameters across all tested models. We use temperature $T = 0.7$, top-k $= 50$ and top-p $= 1.0$. For chunk-level beam search, the parameters are configured as follows: beam width $W = 2$, successors per state $K = 2$, and chunk length $L = 30$. To maintain computational fairness, we set the number of samples N to 4 for the BoN sampling. For EFT, we report the results of fixed β_e, β_i, which are the best parameters evaluated on `Llama-2-7b-chat-hf`.

A.2.3 Compute Resources Specification.

Models are evaluated on 805 test prompts. Model inference takes place on one single NVIDIA A100 GPU for 7B models and on four for others.

B Extended Experimental Results

Table 3 and Table 4 present complete experimental results under the instruction following task. EFT (Mitchell et al., 2023) represents applying token-wise sampling during generation and CBS (Zhou et al., 2024c) represents applying chunk-level beam search during generation. The suffix "i" and "e" indicates that implicit and explicit value functions. Note that the proposed Integrated Value Guidance (IVG) is shown as EFTi (β_i),CBSe.

In addition to `gpt-4-turbo` evaluations, we evaluate response by using two top-rank reward models from RewardBench (Lambert et al., 2024b): `UltraRM-13b` (Cui et al., 2023) and `Starling-RM-34B` (Zhu et al., 2023). SRM denotes the scores evaluated by `Starling-RM-34B` and URM denotes the scores evaluated by `UltraRM-13b`.

Models	SRM (↑)	URM (↑)	LC WR (%)	WR (%)
Tulu Guidance				
`Llama-2-7b-chat`				
Base	−5.83	1.24	10.04	10.16
EFTe (β_e)	−5.82	1.26	N/A	N/A
EFTi (β_i)	−5.68	1.53	10.09	10.79
BoNe	−5.53	2.04	12.00	12.82
CBSi	−5.61	1.78	11.96	13.06
EFTi (β_i),CBSi	−5.56	1.86	11.76	12.82
CBSe	−5.30	2.58	13.76	15.01
EFTi (β_i),CBSe	**-5.18**	**2.95**	**16.46**	**18.00**
`Llama-2-70b-chat`				
Base	−5.61	1.94	16.93	15.71
EFTe (β_e)	−5.57	1.87	N/A	N/A
EFTi (β_i)	−5.47	2.16	16.07	15.44
BoNe	−5.30	2.58	18.70	18.38
CBSi	−5.43	2.29	18.57	17.97
EFTi (β_i),CBSi	−5.33	2.50	17.38	17.40
CBSe	−5.26	2.79	19.32	19.06
EFTi (β_i),CBSe	**-5.11**	**3.08**	**20.72**	**21.26**
`Mistral-7B-Instruct-v0.2`				
Base	−5.72	2.05	19.51	16.27
EFTe (β_e)	−11.12	−8.01	N/A	N/A
EFTi (β_i)	−5.64	2.27	19.62	16.52
BoNe	−5.42	2.89	22.10	18.79
CBSi	−5.47	2.74	22.77	19.34
EFTi (β_i),CBSi	−5.43	2.77	20.47	19.02
CBSe	−5.19	3.54	**27.50**	24.21
EFTi (β_i),CBSe	**-5.05**	**3.70**	26.51	**25.15**
`Mixtral-8x7B-Instruct-v0.1`				
Base	−5.67	1.89	25.58	19.56
EFTe (β_e)	−8.85	−3.58	N/A	N/A
EFTi (β_i)	−5.55	2.05	25.32	20.25
BoNe	−5.33	2.68	28.71	23.61
CBSi	−5.43	2.38	28.62	22.85
EFTi (β_i),CBSi	−5.33	2.56	27.67	22.50
CBSe	−5.14	3.14	32.49	27.69
EFTi (β_e),CBSe	**-5.12**	**3.21**	**33.75**	**28.30**

Table 3: **Instruction following performance under the Tulu guidance.** $\beta_i, \beta_e = 1.0, 1.0$. All CBS shows the results with $W, K, L = 2, 2, 30$. BoN shows the results with $N = 4$.

Models	SRM (↑)	URM (↑)	LC WR (%)	WR (%)
Ultra Guidance				
`Llama-2-7b-chat`				
Base	−5.83	1.24	10.04	10.16
EFTe (β_e)	−5.81	1.30	N/A	N/A
EFTi (β_i)	−5.64	1.68	11.53	12.04
BoNe	−5.61	1.83	12.51	12.78
CBSi	−5.55	1.81	12.36	13.04
EFTi (β_i),CBSi	−5.42	2.23	13.58	14.49
CBSe	−5.47	2.19	13.31	13.81
EFTi (β_i),CBSe	**-5.29**	**2.71**	**15.92**	**16.70**
`Llama-2-70b-chat`				
Base	−5.61	1.94	16.93	15.71
EFTe (β_e)	−5.62	1.89	N/A	N/A
EFTi (β_i)	−5.49	2.16	17.85	17.07
BoNe	−5.39	2.40	18.58	17.68
CBSi	−5.35	2.47	18.20	17.70
EFTi (β_i),CBSi	−5.29	2.60	19.21	18.70
CBSe	−5.36	2.57	19.82	18.54
EFTi (β_i),CBSe	**-5.25**	**2.84**	**21.69**	**21.06**
`Mistral-7B-Instruct-v0.2`				
Base	−5.72	2.05	19.51	16.27
EFTe (β_e)	−7.36	−0.85	N/A	N/A
EFTi (β_i)	−5.62	2.25	19.64	17.70
BoNe	−5.55	2.64	24.04	19.18
CBSi	−5.44	2.81	22.78	19.70
EFTi (β_i),CBSi	−5.36	2.88	21.71	20.49
CBSe	−5.38	3.12	**26.78**	22.23
EFTi (β_i),CBSe	**-5.26**	**3.36**	26.46	**23.60**
`Mixtral-8x7B-Instruct-v0.1`				
Base	−5.67	1.89	25.58	19.56
EFTe (β_e)	−5.78	1.74	N/A	N/A
EFTi (β_i)	−5.59	2.02	25.11	20.25
BoNe	−5.43	2.46	29.29	23.35
CBSi	−5.40	2.54	28.76	23.09
EFTi (β_i),CBSi	−5.33	2.59	29.92	**25.27**
CBSe	−5.33	2.78	**31.57**	25.18
EFTi (β_i),CBSe	**-5.30**	**2.78**	29.99	24.24

Table 4: **Instruction following performance under the Ultra guidance.** $\beta_i, \beta_e = 1.5, 1.0$. All CBS shows the results with $W, K, L = 2, 2, 30$. BoN shows the results with $N = 4$.