REAR: SCALABLE TEST-TIME PREFERENCE REALIGNMENT THROUGH REWARD DECOMPOSITION

Anonymous authors
Paper under double-blind review

ABSTRACT

Aligning large language models (LLMs) with diverse user preferences is a critical yet challenging task. While post-training methods can adapt models to specific needs, they often require costly data curation and additional training. Test-time scaling (TTS) presents an efficient, training-free alternative, but its application has been largely limited to verifiable domains like mathematics and coding, where response correctness is easily judged. To extend TTS to the domain of preference alignment, we introduce a novel framework that models the task as a realignment problem, as the base model often fails to sufficiently align with the preference. Our key insight is to decompose the underlying reward function into two components: one related to the question and the other to user preference. This allows us to derive a REAlignment Reward (REAR) that selectively rescales the preference-related reward while preserving the question-related reward. We show that REAR can be formulated as a linear combination of policy probabilities, making it computationally efficient and easy to integrate with existing TTS algorithms like best-of-N sampling and tree-search algorithms. Experiments on various preference alignment and role-playing benchmarks demonstrate that TTS with REAR enables scalable and effective test-time realignment with superior performance.

1 INTRODUCTION

The remarkable success of Large Language Models (LLMs) in aligning with human preferences is largely attributed to techniques such as Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022; Bai et al., 2022; Rafailov et al., 2023; Guo et al., 2025). This alignment enables a wide range of applications, from personalized assistants (OpenAI, 2023; Chen et al., 2024; Cui et al., 2024) to recommendation systems (Wu et al., 2024; Xue et al., 2023). However, a fundamental challenge remains: the preference alignment of a pretrained model is inherently tied to its training data. This often leads to a mismatch when the model is applied to downstream tasks that require personalized or diverse preferences (Jang et al., 2023; Zhang et al., 2025b;d). While this gap can be bridged through task-specific post-training (Zhang et al., 2025b; Li et al., 2025b), such methods demand significant investment in data curation and computational resources.

To circumvent the costs of post-training, we explore aligning models at inference time. While some approaches modify the policy distribution at the token level to reflect user preferences (Zhang et al., 2025c; Gao et al., 2024), they tend to be computationally intensive and scale poorly. A more promising direction is Test-Time Scaling (TTS) (OpenAI, 2024; Muennighoff et al., 2025; Beeching et al., 2025), where models leverage additional computation during generation to enhance output quality. However, existing TTS research has predominantly focused on domains such as mathematics and coding, where the correctness can be easily verified (OpenAI, 2024). Applying TTS to preference alignment is more challenging, as the quality of a response is holistic and not reducible to a simple verifiable answer. This raises a critical question: how can we effectively guide a TTS framework to evaluate and improve responses for complex preference alignment tasks?

In this work, we address this challenge by framing the TTS process as a realignment problem. We posit that while a pretrained model possesses general instruction-following abilities, its original training objective may not be optimal for a specific user's needs. An inference-time realignment process can rescale the importance of user preference to generate a more aligned response. As illustrated in Figure 1, when a user asks for enjoyable ways to study math but expresses a dislike

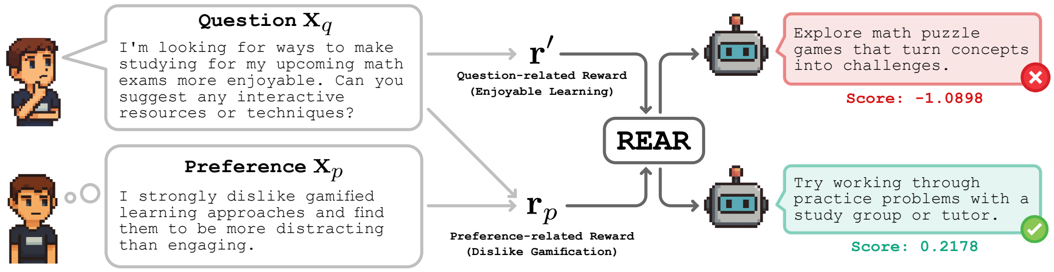


Figure 1: A motivating example of REAR. The method realigns its response from a gamified suggestion to a collaborative one when selecting candidate responses according to REAR scores.

for gamification, a TTS method might generate multiple responses. Some responses may only focus on answering the question of "enjoyable learning approaches", while the preferred responses should also align with the preference on "dislike gamification". Our REAlignment Reward (REAR) is designed to capture the preference alignment capabilities. Specifically, we decompose the reward of a pretrained LLM into a question-related component and a preference-related component. REAR then rescales the preference component to acquire a realigned reward value, allowing us to score the candidate responses and thus effectively select the most aligned option. We further show that REAR can be efficiently computed as a linear combination of policy probabilities, and then incorporate REAR into two TTS methods: a simple best-of-N sampling strategy (Stiennon et al., 2020) and a more sophisticated tree-search algorithm DVTS (Beeching et al., 2025). The contributions of this paper are summarized as follows:

- We formalize test-time preference alignment as a realignment problem and propose REAR, a computationally efficient reward from a decomposed preference alignment objective.
- We develop two scalable TTS methods guided by REAR: a best-of-N sampling approach and a DVTS-based search algorithm.
- Extensive experiments on preference alignment and role-play benchmarks show that our REAR-guided TTS methods outperform existing test-time alignment approaches.

2 Preliminaries

In this section, we first formalize the text generation problem as a Markov Decision Process (MDP) (Puterman, 1994; Sutton & Barto, 2018) at the token level. The MDP model enables us to see how we can apply modern reinforcement learning (RL) algorithms (Schulman et al., 2017; 2015) to text generation problems. Then we will provide a view of reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022) from the perspective of rewards in the given MDP model.

2.1 Token-level MDP for Text Generation

We can model the text generation process as an MDP according to Ramamurthy et al. (2023). The MDP can be defined as a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma, \rho, T \rangle$, where $\mathcal{S}$ is the state space and $\mathcal{A}$ is the action space defined as the vocabulary of a language model, where each action is a token in the vocabulary. We use $\pi(a \mid s)$ to denote the policy, i.e., an LLM, that provides a distribution of actions given the state s. At the beginning of text generation, the prompt $x = (x_1, x_2, \cdots, x_m)$ of length m is sampled from the initial distribution $\rho(s)$ as the initial state s_0, while we use the policy $\pi(\cdot \mid s_t)$ to sample an action a_t at each time step $t \in \{1, \ldots, T\}$. The MDP thus transits to the next state $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$ according to the transition function $\mathcal{P}$. The transition function is a deterministic function satisfying $\mathcal{P}(s_{t+1} \mid s_t, a_t) = 1$ when $s_{t+1} = s_t \oplus a_t$, where $\oplus$ is the concatenation operation. The reward function $r(s_t, a_t)$ is given at each time step t, where the model maximizes the discounted cumulative reward with a discount factor γ. The episode terminates when the model generates an end-of-sequence token defined in the vocabulary or exceeds the maximum length T. We assume that the early-stop sequence is also padded to length T for notational simplicity.

2.2 Reinforcement Learning from Human Feedback

The main objective of reinforcement learning from human feedback (RLHF) is to find a policy that can maximize the expected cumulative reward of the defined MDP. Classical RLHF methods (Ouyang et al., 2022; Bai et al., 2022) usually consider learning a reward model to turn human preferences into reward signals. The objective can be formulated as follows:

$$\max_{\pi} \mathbb{E}_{s_0\sim\rho, a_t\sim\pi(\cdot|s_t), s_{t+1}\sim\mathcal{P}(s_t,a_t)} \left[\sum_{t=0}^{T} \gamma^t (r(s_t,a_t) - \beta D_{\mathrm{KL}}(\pi(\cdot|s_t) \| \pi_{\mathrm{ref}}(\cdot|s_t))) \right], \tag{1}$$

where we use D_{KL} to denote the Kullback-Leibler (KL) divergence and β is a hyper-parameter to limit the divergence between the policy to be learned π and a reference policy π_{ref}. The reference policy usually comes from the base model that is used to initialize RL training. Following Li et al. (2025b), we can convert this objective from the perspective of maximum entropy RL (Haarnoja et al., 2018) according to the following proposition.

Proposition 2.1. *The optimization problem in Equation* (1) *is equivalent to*

$$\max_{\pi} \mathbb{E}_{s_0\sim\rho} \left[\mathbb{E}_{a_0\sim\pi(\cdot|s_0)} [Q^\pi(s_0,a_0) + \beta\mathcal{H}(\pi(\cdot|s_0))] \right], \tag{2}$$

where $\mathcal{H}(\pi(\cdot|s_t)) = \mathbb{E}_{a_t\sim\pi(\cdot|s_t)}[-\log\pi(a_t|s_t)]$ *is the entropy of* π *in the state* s_t*, and*

$$Q^\pi(s_0,a_0) = \mathbb{E}_{s_t\sim\mathcal{P}(s_0,a_0), a_t\sim\pi(\cdot|s_t)} \left[r'(s_0,a_0) + \sum_{t=1}^{T} \gamma^t (r'(s_t,a_t) + \beta\mathcal{H}(\pi(\cdot|s_t))) \right]. \tag{3}$$

is the soft-Q function of the policy π*. The reshaped reward* $r'(s,a) = r(s,a) + \beta\log\pi_{\mathrm{ref}}(a|s)$*. The soft-Q function* Q^π *satisfies the following Bellman equation:*

$$Q^\pi(s,a) = r'(s,a) + \gamma\mathbb{E}_{s'\sim\mathcal{P}(s,a), a'\sim\pi(\cdot|s')}[Q^\pi(s',a') + \beta\mathcal{H}(\pi(\cdot|s'))] = r'(s,a) + \gamma V^\pi(s'). \tag{4}$$

We denote $V^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}[Q^\pi(s,a) + \beta\mathcal{H}(\pi(\cdot|s))]$ *as the value function of policy* π*.*

We defer the proof to Appendix A.1. Here Proposition 2.1 shows that the RLHF objective can be converted to the maximum entropy RL problem under the reward r'. As this optimization manner is widely used in LLM research, we can thus use various open-source LLMs to address our preference realignment problem described in the following section.

3 Test-time Realignment through Reward Decomposition

In this section, we detail our method for test-time preference realignment. We begin by introducing the theoretical foundation of our approach: a reward decomposition that separates the model's objective into question-related and preference-related components. Based on this, we derive our REAlignment Reward (REAR), a score that allows us to control the emphasis on user preference. Finally, we show how REAR can be integrated into standard test-time scaling (TTS) algorithms like best-of-N sampling (Stiennon et al., 2020) and DVTS (Beeching et al., 2025) to produce more aligned responses.

3.1 Realignment Reward (REAR)

In a preference alignment task, an LLM receives a question prompt x_q and a preference prompt x_p. The concatenated prompt $x = x_q \oplus x_p$ is used to generate a response. To formalize this into a token-level MDP form, we define the state s as the sequence that contains the full prompt x and the generated answer, and the state s^q as the sequence that contains only the question and the generated answer. Therefore, we can obtain two reward terms $r'(s,a)$ and $r'(s^q,a)$, which represent the reward when considering the full prompt and only the question part, respectively. Although we cannot directly access these rewards, there exists a relationship between these two terms. Intuitively, the reward $r'(s,a)$ should contain both the reward $r'(s^q,a)$ which only considers the question, and an additional reward that focuses on the preference part, which forms the following equation.

$$r'(s,a) = r'(s^q,a) + \alpha r_p(s,a), \tag{5}$$

where $r_p(s,a)$ is a preference-related reward that reflects how the chosen action aligns with the given preference. Here we introduce a linear combination to decompose the reward $r'(s,a)$ into the question-related reward $r'(s^q,a)$ and the preference-related reward $r_p(s,a)$.

Lemma 3.1. *The policy $\pi(a|s)$ when taking the full prompt x as input is the optimal policy of the following optimization problem under the decomposition in Equation* (5)*:*

$$\max_{\hat{\pi}} \mathbb{E}_{s_0\sim\rho, a_t\sim\hat{\pi}(\cdot|s_0), s_{t+1}\sim\mathcal{P}(s_t,a_t)} \left[\sum_{t=0}^{T} \gamma^t \left(r_p(s_t,a_t) - \frac{\beta}{\alpha} D_{\mathrm{KL}}(\hat{\pi}(\cdot|s_t) \| \pi(\cdot|s_t^q)) \right)\right], \tag{6}$$

where s_t^q is the corresponding question-only state of s_t.

The proof can be found in Appendix A.2. Lemma 3.1 reveals that the original policy $\pi(a|s)$ implicitly maximizes the preference-related reward $r_p(s,a)$ subject to a constraint on the KL-divergence from the distribution of the question-only policy. This framing of Reward Decomposition is essential. Unlike heuristic strategies such as simple policy interpolation, proving that the base model inherently optimizes a specific reward structure allows us to treat the derived REAR score as a valid value function. This theoretical foundation validates the use of lookahead search algorithms like DVTS, which require a consistent reward signal, rather than being limited to simple sampling heuristics. This framing suggests a clear path to realignment: if we could control this trade-off at test time, we could steer the generation to be more or less aligned with the preference. To this end, we introduce a new, flexible coefficient $\hat{\alpha}$ to re-weight the preference component, defining our realignment reward as:

$$r_{\mathrm{REAR}}(s,a) = r'(s^q,a) + \hat{\alpha} r_p(s,a). \tag{7}$$

By adjusting $\hat{\alpha}$ at test time, we can modulate the influence of the preference reward, steering the generation towards responses that are more aligned with a user's specific needs, without altering the underlying model. The challenge here is that $r_{\mathrm{REAR}}(s,a)$ is defined in terms of unobserved reward components. Fortunately, the framework of maximum entropy RL (Haarnoja et al., 2018; Li et al., 2025a) allows us to express this reward in a computable form based on policy probabilities.

Lemma 3.2. *The realignment reward $r_{\mathrm{REAR}}(s,a)$ keeps policy-optimality with the following proxy reward:*

$$\hat{r}_{\mathrm{REAR}}(s,a) = \frac{(\alpha-\hat{\alpha})\beta}{\alpha} \log \pi(a \mid s^q) + \frac{\hat{\alpha}\beta}{\alpha} \log \pi(a \mid s). \tag{8}$$

Intuitively, this substitution is grounded in Maximum Entropy RL, where the optimal policy follows a Boltzmann distribution $\pi^*(a|s) \propto \exp(Q^*(s,a)/\beta)$. Since the Q-function represents the long-term cumulative reward, the log-probability of the policy is directly proportional to the reward plus value function terms. This allows us to mathematically recover the implicit reward optimizing the policy from the log-probabilities themselves, providing a dense, token-level signal without training a separate reward model. We defer the detailed proof to Appendix A.3, which shows that the difference between the two rewards is a potential-based shaping term (Ng et al., 1999).

3.2 Test-time Scaling with REAR

Our goal is to find a policy that maximizes the expected discounted REAR at inference time. According to Lemma 3.2, this is equivalent to maximizing the expected discounted proxy reward $\hat{r}_{\mathrm{REAR}}(s,a)$. Since the optimal policy is invariant to positive scaling of the reward function, we can simplify $\hat{r}_{\mathrm{REAR}}(s,a)$ by omitting the constant factor β to derive the following score function:

$$S(s,a) = (1-\lambda)\log \pi(a \mid s^q) + \lambda \log \pi(a \mid s), \tag{9}$$

where we set $\lambda = \frac{\hat{\alpha}}{\alpha} > 0$ as a hyper-parameter. This concise formulation reveals how we integrate the LLM preferences that are hard to verify by encoding its output probability to a token-level reward. Intuitively, $\lambda > 1$ indicates that the preference is more important in the real case than when the model is trained and $\lambda < 1$ will reduce the importance of the preference. When $\lambda = 1$, the result is equivalent to directly using the original LLM for inference. In our experiments, we find that choosing a relatively large λ will yield better performance on benchmark scores in most tasks.

This score can be extended to a response trajectory $\tau = (s_0, a_0, \ldots, s_T, a_T)$ across multiple tokens in the form of a cumulative score:

$$S(\tau) = \sum_{t=0}^{T} \gamma^t S(s_t,a_t). \tag{10}$$

Since τ can represent either a complete or partial response, this formulation allows for flexible integration with various TTS methods. We explore two such methods:

Best-of-N (BoN) with REAR. We simply sample N responses and calculate the REAR score for each response. Then we select the response with the highest score as the final response.

Diverse Verifier Tree Search (DVTS) with REAR. We use the DVTS (Beeching et al., 2025) algorithm to select a final response, where the response generated step-by-step in a tree search manner and selected according to the REAR score.

Compared to external or generative reward models (Lambert et al., 2025; Liu et al., 2024a; Zhang et al., 2024; Mahan et al., 2024), REAR solves the preference alignment problem by solely rescaling its inherent preferences, without requiring extra training, external model calls or extra generation steps. This makes REAR highly flexible and readily deployable in a plug-and-play manner across almost any LLM. Moreover, since REAR provides a token-level reward formulation, it can be applied to partial responses, enabling its use with advanced TTS algorithms like DVTS, which is not valid for general reward models that can only perform effective evaluations with the whole response.

4 Related Work

Preference Alignment Aligning LLMs with human preferences is a central challenge in AI safety and usability. Early and prominent approaches rely on training-based methods, particularly reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022; Bai et al., 2022), where a reward model is trained on human preference data to fine-tune a base model. Subsequent work has sought to simplify this pipeline (Rafailov et al., 2023) bypassing the need for an explicit reward model. Other approaches focus on creating specialized data curricula (Zhang et al., 2025b) or maintaining original capabilities when adapting to new preferences (Li et al., 2023; Wang et al., 2025; Li et al., 2025b). While effective, these training-based methods often require extensive data and are computationally expensive. This motivates a shift towards test-time alignment methods that adapt model behavior without updating weights. For instance, Zhang et al. (2025c) and Gao et al. (2024) propose techniques to modify the model's output distribution at each generation step to better align with given preferences. Our work builds on this line of research but focuses on scaling the alignment process through a novel reward formulation within a TTS framework rather than direct policy modification, which provides a stable and scalable performance improvement.

Test-time Scaling Test-time scaling (TTS) aims to improve model performance by allocating more computational resources during inference, realized by extended thinking (OpenAI, 2024; Guo et al., 2025; Muennighoff et al., 2025) or parallel searching (Wang et al., 2024a; Comanici et al., 2025; Huang & Yang, 2025). This paradigm has been particularly successful in domains where answers can be easily extracted and verified, such as mathematical and coding problems (OpenAI, 2024; Zhang et al., 2025a), where researchers adopt self-consistency (Wang et al., 2023; Li et al., 2024) and use explicit verifiers such as process-based reward models (Lightman et al., 2024; Wang et al., 2024b) with sophisticated search algorithms (Wei et al., 2022; Yao et al., 2023; Wang et al., 2024a) that explore different reasoning paths. However, applying TTS to open-ended preference alignment tasks is challenging due to the absence of a simple, verifiable ground truth. Generative reward models (Zhang et al., 2024; Mahan et al., 2024; Liu et al., 2025) are proposed for their ability to verify an answer through the generation process but still face challenges on computational efficiency and accuracy. Recent study (Li et al., 2025a) indicates that the LLM itself is an implicit reward model, supporting the validity of policy probabilities as rewards. Our approach differs by deriving a specialized reward, REAR, that is specifically designed for preference realignment and can be integrated into various TTS algorithms, bridging the gap between TTS for verifiable reasoning and TTS for subjective preference alignment. Unlike methods like ARGS (Khanov et al., 2024) or IVG (Liu et al., 2024b) which rely on training and hosting external Reward Models or value heads, REAR is fully training-free and derives its signal solely from the base model's internal probabilities. This allows REAR to extend TTS to open-ended domains where no ground-truth verifiers exist.

5 Experiments

In this section, we investigate the efficacy of REAR-guided test-time sampling (TTS) on existing preference alignment tasks. We first describe our experimental setup in Section 5.1. Then in Section 5.2, we specifically seek to determine whether our proposed hyperparameter, λ, can effectively control the degree of alignment with user preferences. In Section 5.3, we evaluate the performance of our method against several baselines, including other test-time preference alignment methods and TTS approaches that use different reward forms. In Section 5.4, we further show the scaling performance of our methods and analyze the robustness and efficiency of our method.

5.1 Experimental Setup

Evaluation Benchmarks To evaluate the preference alignment capabilities of different methods, we use three recent benchmarks:

- **PrefEval** (Zhao et al., 2025) requires the LLM to generate personalized responses across conversations according to the user's previously stated preferences, which provides a comprehensive evaluation of the LLM's capability on inferring, remembering, and applying the user preference to multi-turn conversations. The PrefEval benchmark contains three data types, including explicit preference, implicit choice, and implicit preference.
- **Multifaceted Bench** (Lee et al., 2024) is designed to evaluate whether the LLM can generate context-specific responses tailored to user preferences. Each sample is paired with synthetic system messages and reference answers.
- **PingPong** (Gusev, 2024) evaluates the role-playing capabilities of LLMs through a multi-turn conversation. As role-playing can be framed as a preference alignment problem, we use this benchmark to assess our method's effectiveness in this practical scenario.

Baselines Beyond greedy decoding, several methods can align model outputs with human preferences. We compare REAR against baselines from two main categories, with implementation details provided in Appendix C:

- **Test-time preference alignment methods**. We include two representative methods: Amulet (Zhang et al., 2025c) and Linear Alignment (LA) (Gao et al., 2024). These methods align generations with preferences by modifying the token-level generation probability distribution.
- **Test-time Sampling with Other Rewards**. Like our method, these baselines use best-of-N (BoN) sampling but employ different reward sources. We consider two variants: one using an external reward model (External RM) and another using the generative model itself as a reward source (GenRM). For the external RM, we use the Skywork-Reward-Llama-8B model (Liu et al., 2024a) due to its strong performance on RewardBench (Lambert et al., 2025) and its comparable size to our base model.

For our main experiments, we use `Qwen2.5-7B-Instruct` (Yang et al., 2024) as the base model. We employ the SGLang inference engine (Zheng et al., 2024) for response generation, maintaining consistent sampling parameters across all methods except for Amulet and Linear Alignment, for which we use the authors' original implementation (Zhang et al., 2025c). We use $N = 16$ samples for BoN methods in our experiments or equivalent sampling size for DVTS. Further implementation details are provided in Appendix C.

5.2 Controllable Realignment with λ

As established in our methodology, the hyperparameter λ governs the strength of preference alignment by scaling the preference-related reward. A larger λ directs more attention to this reward while a smaller λ may not sufficiently align the model with user preferences but focuses more on answering the question. In this section, we investigate the impact of λ on benchmark performance. We conduct experiments on the PrefEval benchmark, evaluating both Best-of-N and DVTS with REAR across a range of λ values.

We focus on two data types from PrefEval: explicit preference and implicit choice. Although derived from the same source data, they employ different prompting and evaluation protocols. For the explicit preference task, the model must generate a response that adheres to a given system preference prompt. An external LLM judge evaluates the response quality based on multiple rubrics, including helpfulness, preference violation, consistency, and hallucination. We report the average score across

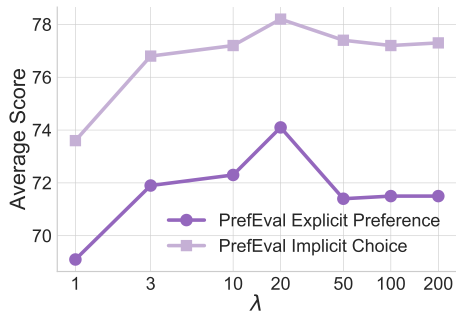


Figure 2: Benchmark scores of REAR-guided TTS methods on PrefEval explicit preference and implicit choice data with different λ values.

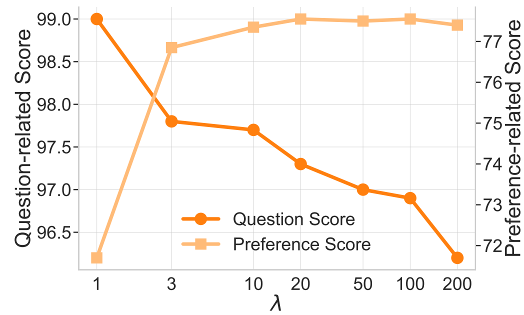


Figure 3: Scores on questions and preference of REAR-guided TTS methods on PrefEval explicit preference data with different λ values.

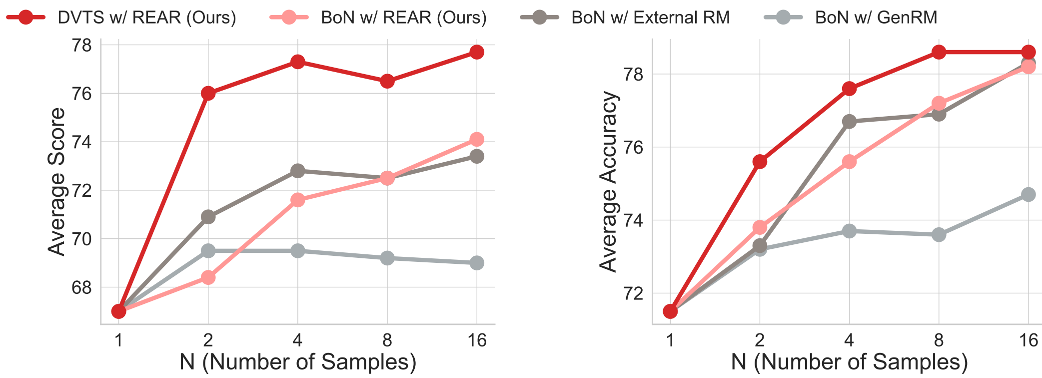
Table 1: Performance comparison of REAR-guided TTS methods and other baselines on various preference alignment benchmarks. Bold values indicate the best performance on the corresponding benchmark.

Benchmark	DVTS w/ REAR (Ours)	BoN w/ REAR (Ours)	Greedy	External RM	GenRM	Amulet	LA
PrefEval Scores							
Explicit Preference	**77.7**	74.1	67.0	73.4	69.0	68.5	64.2
Implicit Choice	**78.6**	78.2	71.5	78.3	74.7	70.4	78.0
Implicit Preference	**19.1**	16.2	12.0	17.0	12.9	13.1	12.8
Multifaceted Bench	**76.8**	76.3	75.3	76.5	76.1	75.4	75.6
Ping-Pong Bench							
Score	3.03	**3.07**	2.97	2.97	3.01	2.87	3.01
Stay in Character Score	2.19	**2.35**	2.01	2.10	2.09	2.07	2.13
Fluency Score	4.67	4.50	**4.88**	4.52	4.76	4.47	4.70
Entertaining Score	2.24	**2.36**	2.02	2.27	2.18	2.09	2.20

all rubrics. In contrast, the implicit choice task presents preferences within a multi-turn conversation, from which the model must infer the user's inclination. The evaluation is a multiple-choice question where the model selects the most preferred response out of four options, and performance is measured by accuracy.

Benchmark Scores with Different λ As shown in Figure 2, the performance of BoN with REAR on both PrefEval explicit preference and implicit choice data varies with λ. The scores for both data types follow a similar trend: they first increase and then decrease as λ grows. Optimal performance on both tasks is achieved consistently at $\lambda = 20.0$, with lower scores observed for both smaller and larger values of λ. We also find similar trends when adjusting λ in other tasks and the results are deferred to Appendix E.

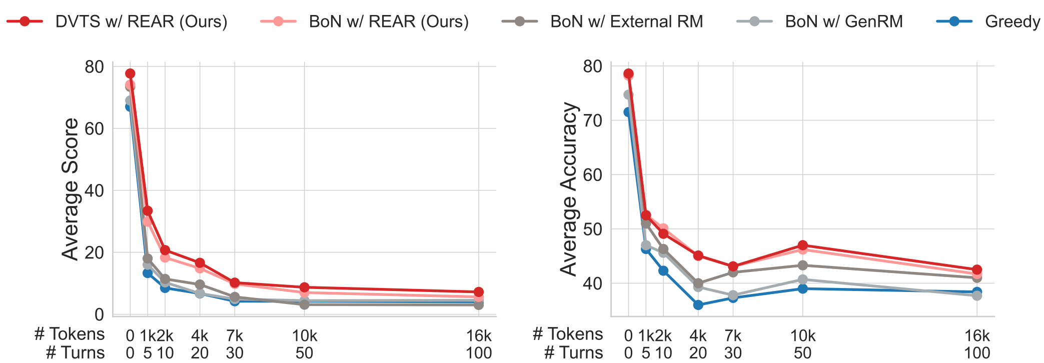
Analysis on Generated Responses To understand this non-monotonic relationship, we analyze how λ affects different aspects of response quality. The detailed rubrics from the PrefEval explicit preference task allow us to disentangle performance into two components: general response quality and preference alignment. We use the "helpfulness" score to measure the former and the average of "preference violation" and "preference acknowledgement" scores for the latter. As illustrated in Figure 3, these two components exhibit monotonic trends with respect to λ. As λ increases, the preference-related score improves, while the question-related score (helpfulness) declines. This trade-off explains why simply increasing λ does not guarantee better overall performance; an excessively large λ compromises the model's fundamental ability to provide helpful answers, thereby reducing the overall quality of the response.



(a) PrefEval explicit preference

(b) PrefEval implicit choice

Figure 4: Scaling performance on the PrefEval benchmark with varying numbers of samples (N) for different methods. We use the average LLM-evaluated scores for the explicit preference task (left) and the accuracy of selected choices for the implicit choice task (right).



(a) PrefEval explicit preference

(b) PrefEval implicit choice

Figure 5: Long-context performance of REAR-guided TTS methods and other baselines on the explicit preference data and implicit choice data from the PrefEval benchmark with augmented conversation turns. The x-axis indicates the number of conversation turns and the estimated total number of tokens for the augmented conversational data. We use the average LLM-evaluated scores for the explicit preference task (left) and the accuracy of choices for the implicit choice task (right).

5.3 Performance Comparisons

We compare our methods against the baselines on the PrefEval, Multifaceted, and Ping-Pong benchmarks. As shown in Table 1, both BoN with REAR and DVTS with REAR outperform all baselines on most benchmarks, demonstrating strong performance on both accuracy-based (PrefEval implicit choice) and LLM-evaluated tasks. The BoN baseline using an external RM also performs competitively, likely because the external model provides a valuable additional reward signal to select the best response. In contrast, using the generative model itself as a reward model (GenRM) does not yield significant improvement, suggesting that the model struggles to reliably verify its own responses. In addition, the test-time preference alignment methods, including Amulet and LA, also underperform on these benchmarks.

On a benchmark-specific level, we observe a significant performance drop on the PrefEval implicit preference task compared to the other two PrefEval tasks, which is consistent with previous findings (Zhao et al., 2025). Interestingly, on the Ping-Pong benchmark, all TTS methods achieve higher scores on the “stay-in-character” and “entertaining” rubrics, but decrease the “fluency” score, where greedy decoding performs best. This suggests that TTS methods prioritize role-playing traits at the expense of fluency. For the multifaceted benchmark, while we do not report detailed rubric scores since they differ from specific samples, our DVTS variant again outperforms the baselines.

5.4 Robustness and Efficiency of REAR-guided TTS

Scaling Performance We investigate how the performance of our method scales with the number of samples (N). As shown in Figure 4, performance on the PrefEval explicit preference and implicit choice datasets improves as N increases, with diminishing returns for larger values. Our BoN approach with REAR demonstrates scaling performance comparable to the variant using an external RM. The DVTS variant achieves stronger performance with a smaller sampling budget, highlighting the efficiency of its step-by-step tree search approach.

Robustness on Long-context Input A key advantage of REAR is that it is derived directly from the generation process, making it inherently robust. This becomes particularly evident with out-of-distribution inputs, such as long-context prompts. Following the methodology from Zhao et al. (2025), we evaluate robustness by augmenting conversations with additional turns inserted between the preference context and the question. As shown in Figure 5, our methods consistently outperform the baselines across various context lengths. Test-time alignment baselines including Amulet and LA are excluded due to out-of-memory errors on long-context data. The performance of BoN with an external RM and GenRM degrades significantly on long-context inputs, occasionally falling below the greedy baseline, since the augmented data lies outside the external RM's training distribution, leading to unreliable reward signals.

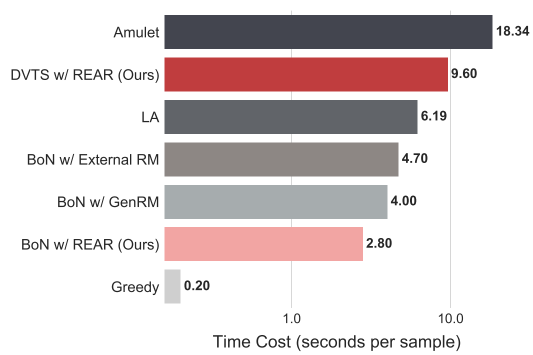


Figure 6: Time cost of different methods on the PrefEval explicit preference task.

Efficiency of REAR REAR offers significant efficiency gains over baselines that rely on external reward models. We report the inference cost of REAR-guided methods compared to other baselines on the PrefEval explicit preference task in Figure 6, using a node of 8 NVIDIA GPUs with 96GB memory, by calculating rewards from the model's internal probabilities, REAR avoids the substantial computational overhead of loading and executing additional models. This makes REAR-guided methods not only more efficient but also easier for deployment.

6 Conclusion

In this work, we introduced the REAlignment Reward (REAR), a novel and efficient reward that realigns LLM to user preferences at test time. By decomposing the underlying reward into question-related and preference-related portions, we can calculate REAR directly from the model's own policy probabilities. We further integrate two test-time scaling methods, best-of-N sampling and DVTS, into REAR, enabling controllable and effective preference realignment without any model training. Extensive experiments show that REAR-guided TTS methods significantly outperforms both existing test-time alignment techniques and TTS methods guided by other rewards across a range of preference alignment benchmarks. Our work provides a controllable and scalable solution for personalizing LLM interactions and enables test-time scaling to more subjective, open-ended domains without the need of other models.

Despite these promising results, our work has several limitations. First, the performance of REAR is dependent on the hyperparameter λ, which may differ from data samples. Although we show that the optimal range of λ is relatively consistent, pre-evaluation on a validation dataset or selecting appropriate values with heuristic methods can help further improve the performance. Second, while TTS is more lightweight than fine-tuning a model, it still introduces significant computational overhead at inference time. Identifying the sweet spot of REAR-guided TTS methods without incurring excessive computational cost can be a promising direction.

Ethics Statement

The authors of this paper have adhered to the ICLR Code of Ethics. Our work focuses on improving the alignment of large language models with user preferences, which we believe is a crucial step toward developing safer and more helpful AI systems. We acknowledge that, like any alignment technique, our method could potentially be misused to align models with harmful or unethical preferences. However, the core principles of our approach are designed to provide controllable and transparent realignment, which can also serve as a tool for safety researchers to better understand and mitigate undesirable model behaviors. The experiments are conducted on publicly available benchmarks, which do not contain personally identifiable information. We encourage responsible use of this technology and further research into robust safety guardrails for preference alignment techniques. Our use of large language models for evaluation was conducted via standard APIs, and we acknowledge the associated computational and environmental costs.

Reproducibility Statement

The code and data are provided in the supplementary material, while our used model is publicly available. The README file within the code submission contains detailed instructions on setting up the environment and running experiments presented in the paper. Appendix C also provides a comprehensive description of the implementation details, including the base model used, key hyperparameters, and the setup for all baseline methods. Appendix D details the evaluation protocols for each benchmark.

References

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosiute, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemí Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional AI: Harmlessness from AI feedback. *CoRR*, abs/2212.08073, 2022.

Edward Beeching, Lewis Tunstall, and Sasha Rush. Scaling test-time compute with open models, 2025. URL `https://huggingface.co/spaces/HuggingFaceH4/blogpost-scaling-test-time-compute`.

Jiangjie Chen, Xintao Wang, Rui Xu, Siyu Yuan, Yikai Zhang, Wei Shi, Jian Xie, Shuang Li, Ruihan Yang, Tinghui Zhu, Aili Chen, Nianqi Li, Lida Chen, Caiyu Hu, Siye Wu, Scott Ren, Ziquan Fu, and Yanghua Xiao. From persona to personalization: A survey on role-playing language agents. *Transactions on Machine Learning Research*, 2024, 2024.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit S. Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, Krishna Haridasan, Ahmed Omran, Nikunj Saunshi, Dara Bahri, Gaurav Mishra, Eric Chu, Toby Boyd, Brad Hekman, Aaron Parisi, Chaoyi Zhang, Kornraphop Kawintiranon, Tania Bedrax-Weiss, Oliver Wang, Ya Xu, Ollie Purkiss, Uri Mendlovic, Ilaï Deutel, Nam Nguyen, Adam Langley, Flip Korn, Lucia Rossazza, Alexandre Ramé, Sagar Waghmare, Helen Miller, Nathan Byrd, Ashrith Sheshan, Raia Hadsell Sangnie Bhardwaj, Pawel Janus, Tero Rissa, Dan Horgan, Sharon Silver, Ayzaan Wahid, Sergey Brin, Yves Raimond, Klemen Kloboves, Cindy Wang, Nitesh Bharadwaj Gundavarapu, Ilia Shumailov, Bo Wang, Mantas Pajarskas, Joe Heyward, Martin Nikoltchev, Maciej Kula, Hao Zhou, Zachary Garrett, Sushant Kafle, Sercan Arik, Ankita Goel, Mingyao Yang, Jiho Park, Koji Kojima, Parsa Mahmoudieh, Koray Kavukcuoglu, Grace Chen, Doug Fritz, Anton Bulyenov, Sudeshna Roy, Dimitris Paparas, Hadar Shemtov, Bo-Juen Chen, Robin Strudel, David Reitter, Aurko Roy, Andrey Vlasov, Changwan Ryu, Chas Leichner, Haichuan Yang, Zelda Mariet, Denis Vnukov, Tim Sohn, Amy Stuart, Wei Liang, Minmin Chen, Praynaa Rawlani, Christy Koh, JD Co-Reyes, Guangda Lai, Praseem Banzal, Dimitrios Vytiniotis, Jieru Mei, and Mu Cai. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *CoRR*, abs/2507.06261, 2025.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. ULTRAFEEDBACK: boosting language models with scaled AI feedback. In *International Conference on Machine Learning*, 2024.

Songyang Gao, Qiming Ge, Wei Shen, Shihan Dou, Junjie Ye, Xiao Wang, Rui Zheng, Yicheng Zou, Zhi Chen, Hang Yan, Qi Zhang, and Dahua Lin. Linear alignment: A closed-form solution for aligning human preferences without tuning and feedback. In *International Conference on Machine Learning*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z.

Ilya Gusev. Pingpong: A benchmark for role-playing language models with user emulation and multi-model evaluation. *CoRR*, abs/2409.06820, 2024.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *CoRR*, abs/1812.05905, 2018.

Yichen Huang and Lin F. Yang. Gemini 2.5 pro capable of winning gold at IMO 2025. *CoRR*, abs/2507.15855, 2025.

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

Maxim Khanov, Jirayu Burapacheep, and Yixuan Li. ARGS: Alignment as reward-guided search. In *International Conference on Learning Representations*, 2024.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, Lester James V. Miranda, Bill Yuchen Lin, Khyathi Raghavi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. Rewardbench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics*, pp. 1755–1797, 2025.

Seongyun Lee, Sue Hyun Park, Seungone Kim, and Minjoon Seo. Aligning to thousands of preferences via system message generalization. In *Advances in Neural Information Processing Systems*, 2024.

Chenran Li, Chen Tang, Haruki Nishimura, Jean Mercat, Masayoshi Tomizuka, and Wei Zhan. Residual q-learning: Offline and online policy customization without value. In *Advances in Neural Information Processing Systems*, 2023.

Junyou Li, Qin Zhang, Yangbin Yu, Qiang Fu, and Deheng Ye. More agents is all you need. *CoRR*, abs/2402.05120, 2024.

Yi-Chen Li, Tian Xu, Yang Yu, Xuqin Zhang, Xiong-Hui Chen, Zhongxiang Ling, Ningjing Chao, Lei Yuan, and Zhi-Hua Zhou. Generalist reward models: Found inside large language models. *CoRR*, abs/2506.23235, 2025a.

Yi-Chen Li, Fuxiang Zhang, Wenjie Qiu, Lei Yuan, Chengxing Jia, Zongzhang Zhang, Yang Yu, and Bo An. Q-adapter: Customizing pre-trained llms to new preferences with forgetting mitigation. In *International Conference on Learning Representations*, 2025b.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

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

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

Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. Inference-time scaling for generalist reward modeling. *CoRR*, abs/2504.02495, 2025.

Dakota Mahan, Duy Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. Generative Reward Models. *CoRR*, abs/2410.12832, 2024.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *CoRR*, abs/2501.19393, 2025.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning*, pp. 278–287. Morgan Kaufmann, 1999.

OpenAI. Chatgpt. `https://chat.openai.com/`, 2023. Accessed: 2023-03-14.

OpenAI. Learning to reason with LLMs, 2024. URL `https://openai.com/index/learning-to-reason-with-llms`. Accessed: 2025-05-05.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2023.

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization. In *International Conference on Learning Representations*, 2023.

John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize from human feedback. *CoRR*, abs/2009.01325, 2020.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

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

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Annual Meeting of the Association for Computational Linguistics*, pp. 9426–9439, 2024b.

Pengcheng Wang, Xinghao Zhu, Yuxin Chen, Chenfeng Xu, Masayoshi Tomizuka, and Chenran Li. Residual policy gradient: A reward view of kl-regularized objective. *CoRR*, abs/2503.11019, 2025.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in Language Models. In *International Conference on Learning Representations*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-Thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, Hui Xiong, and Enhong Chen. A survey on large language models for recommendation. *World Wide Web (WWW)*, 27(5):60, 2024.

Wanqi Xue, Qingpeng Cai, Zhenghai Xue, Shuo Sun, Shuchang Liu, Dong Zheng, Peng Jiang, Kun Gai, and Bo An. Prefrec: Recommender systems with human preferences for reinforcing long-term user engagement. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2874–2884, 2023.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *CoRR*, abs/2412.15115, 2024.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of Thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, 2023.

Fuxiang Zhang, Jiacheng Xu, Chaojie Wang, Ce Cui, Yang Liu, and Bo An. Incentivizing llms to self-verify their answers. *CoRR*, abs/2506.01369, 2025a. doi: 10.48550/ARXIV.2506.01369. URL `https://doi.org/10.48550/arXiv.2506.01369`.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative Verifiers: Reward modeling as next-token prediction. *CoRR*, abs/2408.15240, 2024.

Xuemiao Zhang, Liangyu Xu, Feiyu Duan, Yongwei Zhou, Sirui Wang, Rongxiang Weng, Jingang Wang, and Xunliang Cai. Preference curriculum: Llms should always be pretrained on their preferred data. In *Findings of the Association for Computational Linguistics*, pp. 21181–21198, 2025b.

Zhaowei Zhang, Fengshuo Bai, Qizhi Chen, Chengdong Ma, Mingzhi Wang, Haoran Sun, Zilong Zheng, and Yaodong Yang. Amulet: Realignment during test time for personalized preference adaptation of llms. In *International Conference on Learning Representations*, 2025c.

Zhehao Zhang, Ryan A. Rossi, Branislav Kveton, Yijia Shao, Diyi Yang, Hamed Zamani, Franck Dernoncourt, Joe Barrow, Tong Yu, Sungchul Kim, Ruiyi Zhang, Jiuxiang Gu, Tyler Derr, Hongjie Chen, Junda Wu, Xiang Chen, Zichao Wang, Subrata Mitra, Nedim Lipka, Nesreen K. Ahmed, and Yu Wang. Personalization of large language models: A survey. *Transactions on Machine Learning Research*, 2025, 2025d.

Siyan Zhao, Mingyi Hong, Yang Liu, Devamanyu Hazarika, and Kaixiang Lin. Do LLMs recognize your preferences? evaluating personalized preference following in LLMs. In *International Conference on Learning Representations*, 2025.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark W. Barrett, and Ying Sheng. Sglang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems*, 2024.

A DEFERRED PROOFS

A.1 PROOF OF PROPOSITION 2.1

The objective of RLHF is to maximize the expected discounted reward regularized by the KL divergence between the learned policy π and a reference policy π_{ref}:

$$\max_{\pi} \mathbb{E}_{\tau \sim \pi}\left[\sum_{t=0}^{T} \gamma^t (r(s_t, a_t) - \beta D_{\text{KL}}(\pi(\cdot|s_t) \| \pi_{\text{ref}}(\cdot|s_t)))\right], \tag{11}$$

where the expectation is over trajectories $\tau = (s_0, a_0, s_1, \dots)$ sampled from the policy π.

First, we expand the KL divergence term:

$$D_{\text{KL}}(\pi(\cdot|s_t) \| \pi_{\text{ref}}(\cdot|s_t)) = \mathbb{E}_{a_t \sim \pi(\cdot|s_t)}[\log \pi(a_t|s_t) - \log \pi_{\text{ref}}(a_t|s_t)]. \tag{12}$$

Substituting this into the objective and taking the expectation over actions inside the summation gives:

$$\max_{\pi} \mathbb{E}_{\tau \sim \pi}\left[\sum_{t=0}^{T} \gamma^t (r(s_t, a_t) - \beta(\log \pi(a_t|s_t) - \log \pi_{\text{ref}}(a_t|s_t)))\right]. \tag{13}$$

We can rearrange the terms within the summation:

$$\max_{\pi} \mathbb{E}_{\tau \sim \pi}\left[\sum_{t=0}^{T} \gamma^t ((r(s_t, a_t) + \beta \log \pi_{\text{ref}}(a_t|s_t)) - \beta \log \pi(a_t|s_t))\right]. \tag{14}$$

Let us define a reshaped reward function $r'(s_t, a_t) = r(s_t, a_t) + \beta \log \pi_{\text{ref}}(a_t|s_t)$. Additionally, we recognize that the term $-\mathbb{E}_{a_t \sim \pi(\cdot|s_t)}[\log \pi(a_t|s_t)]$ is the entropy of the policy, denoted by $\mathcal{H}(\pi(\cdot|s_t))$. With these substitutions, the objective becomes:

$$\max_{\pi} \mathbb{E}_{\tau \sim \pi}\left[\sum_{t=0}^{T} \gamma^t (r'(s_t, a_t) + \beta \mathcal{H}(\pi(\cdot|s_t)))\right]. \tag{15}$$

This is the standard objective for maximum entropy reinforcement learning. The expected return in this framework is the definition of the soft value function $V^{\pi}(s_0)$. The objective can thus be written in terms of the soft Q-function and entropy at the initial state, which is equivalent to the formulation in Equation (2).

A.2 PROOF OF LEMMA 3.1

Let $\pi_q(\cdot|s) = \pi(\cdot|s^q)$. In maximum entropy reinforcement learning, the optimal policy π^* is related to the soft Q-function by $\log \pi^*(a|s) = (Q^{\pi^*}(s, a) - V^{\pi^*}(s))/\beta$, where $V^{\pi^*}(s)$ is the soft value function. The Q-function satisfies the Bellman equation $Q^{\pi^*}(s, a) = r'(s, a) + \gamma V^{\pi^*}(s')$. Combining these, we can express the reshaped reward as:

$$r'(s, a) = \beta \log \pi^*(a|s) + V^{\pi^*}(s) - \gamma V^{\pi^*}(s'). \tag{16}$$

The term $V^{\pi^*}(s')$ depends on the action a through the next state s'. For this proof, we adopt the common approximation that this value is constant with respect to a, which is reasonable when a single token has a limited impact on the total future reward. Under this approximation, we can apply this relation to our two policies, $\pi(a|s)$ and $\pi_q(a|s)$:

$$r'(s, a) = \beta \log \pi(a|s) + C_1(s), \tag{17}$$

$$r'(s^q, a) = \beta \log \pi_q(a|s) + C_2(s), \tag{18}$$

where $C_1(s)$ and $C_2(s)$ are terms independent of the current action a. Using the reward decomposition from Equation (5), $r'(s, a) = r'(s^q, a) + \alpha r_p(s, a)$, we can substitute the expressions above:

$$\beta \log \pi(a|s) + C_1(s) = \beta \log \pi_q(a|s) + C_2(s) + \alpha r_p(s, a). \tag{19}$$

Rearranging the terms, we find the optimality condition for $\pi(a|s)$:

$$\log \pi(a|s) - \log \pi_q(a|s) = \frac{\alpha}{\beta} r_p(s, a) + \text{terms independent of } a. \tag{20}$$

Now, consider the KL-regularized optimization problem from the lemma. The per-step objective to maximize at a state s is:

$$\max_{\hat{\pi}} \mathbb{E}_{a\sim\hat{\pi}(\cdot|s)}[r_p(s,a) + \gamma V(s')] - \frac{\beta}{\alpha} D_{\mathrm{KL}}(\hat{\pi}(\cdot|s)\|\pi_q(\cdot|s)), \tag{21}$$

where $V(s')$ is the value of the next state. The solution $\hat{\pi}^*$ to this optimization is well-known:

$$\hat{\pi}^*(a|s) \propto \pi_q(a|s) \exp\left(\frac{\alpha}{\beta}(r_p(s,a) + \gamma V(s'))\right). \tag{22}$$

Taking the logarithm, we find the optimality condition for $\hat{\pi}^*$:

$$\log \hat{\pi}^*(a|s) - \log \pi_q(a|s) = \frac{\alpha}{\beta} r_p(s,a) + \text{terms independent of } a. \tag{23}$$

Since the optimality conditions in Equation (20) and Equation (23) are identical, their solutions must be identical. Therefore, $\pi(a|s) = \hat{\pi}^*(a|s)$, which proves the lemma.

A.3 Proof of Lemma 3.2

We start from the definition of the realignment reward from Equation (7):

$$r_{\mathrm{REAR}}(s,a) = r'(s^q,a) + \hat{\alpha} r_p(s,a). \tag{24}$$

From the reward decomposition in Equation (5), we can express the preference-related reward $r_p(s,a)$ as:

$$r_p(s,a) = \frac{1}{\alpha}(r'(s,a) - r'(s^q,a)). \tag{25}$$

Substituting this into the definition of $r_{\mathrm{REAR}}(s,a)$, we get:

$$\begin{aligned} r_{\mathrm{REAR}}(s,a) &= r'(s^q,a) + \frac{\hat{\alpha}}{\alpha}(r'(s,a) - r'(s^q,a)) && (26)\\ &= \left(1 - \frac{\hat{\alpha}}{\alpha}\right) r'(s^q,a) + \frac{\hat{\alpha}}{\alpha} r'(s,a). && (27) \end{aligned}$$

Next, we relate the reshaped rewards $r'(s,a)$ and $r'(s^q,a)$ to their respective optimal policies, $\pi(a|s)$ and $\pi_q(a|s) = \pi(a|s^q)$. In maximum entropy RL, the reshaped reward can be expressed in terms of the optimal policy and the soft value functions:

$$r'(s,a) = \beta \log \pi(a|s) + V^{\pi}(s) - \gamma V^{\pi}(s'), \tag{28}$$

where $s' = s \oplus a$ is the next state. Applying this for both $r'(s,a)$ and $r'(s^q,a)$:

$$r'(s,a) = \beta \log \pi(a|s) + V^{\pi}(s) - \gamma V^{\pi}(s'), \tag{29}$$

$$r'(s^q,a) = \beta \log \pi_q(a|s) + V^{\pi_q}(s) - \gamma V^{\pi_q}(s'). \tag{30}$$

Substituting these into the expression for $r_{\mathrm{REAR}}(s,a)$:

$$\begin{aligned} r_{\mathrm{REAR}}(s,a) = &\left(1 - \frac{\hat{\alpha}}{\alpha}\right)(\beta \log \pi_q(a|s) + V^{\pi_q}(s) - \gamma V^{\pi_q}(s')) \\ &+ \frac{\hat{\alpha}}{\alpha}(\beta \log \pi(a|s) + V^{\pi}(s) - \gamma V^{\pi}(s')). \end{aligned} \tag{31}$$

We can group the terms that depend on the action a and those that depend only on the state s:

$$r_{\mathrm{REAR}}(s,a) = \frac{(\alpha - \hat{\alpha})\beta}{\alpha} \log \pi(a \mid s^q) + \frac{\hat{\alpha}\beta}{\alpha} \log \pi(a \mid s) + Z(s,a), \tag{32}$$

where $Z(s,a)$ contains the value function terms:

$$Z(s,a) = \left(1 - \frac{\hat{\alpha}}{\alpha}\right)(V^{\pi_q}(s) - \gamma V^{\pi_q}(s')) + \frac{\hat{\alpha}}{\alpha}(V^{\pi}(s) - \gamma V^{\pi}(s')). \tag{33}$$

The term $Z(s,a)$ can be rewritten as $\Phi(s) - \gamma\Phi(s')$, where $\Phi(s) = \left(1 - \frac{\hat{\alpha}}{\alpha}\right)V^{\pi_q}(s) + \frac{\hat{\alpha}}{\alpha}V^{\pi}(s)$ is a potential function that depends only on the state s. According to the theory of potential-based reward shaping (Ng et al., 1999), adding a reward of the form $\gamma\Phi(s') - \Phi(s)$ to a base reward function does not change the optimal policy. The term $Z(s,a)$ is the negative of such a potential-based shaping reward. Therefore, the optimal policy for the full reward $r_{\mathrm{REAR}}(s,a)$ is identical to the optimal policy for the proxy reward obtained by removing $Z(s,a)$. This justifies using only the action-dependent terms for our score function, which can be expressed as

$$\hat{r}_{\mathrm{REAR}}(s,a) = \frac{(\alpha - \hat{\alpha})\beta}{\alpha} \log \pi(a \mid s^q) + \frac{\hat{\alpha}\beta}{\alpha} \log \pi(a \mid s). \tag{34}$$

B Declaration on the Use of LLMs

We acknowledge the use of Large Language Models (LLMs) to assist in the preparation of this manuscript. Specifically, LLMs were utilized for the following tasks: (1) generating boilerplate code for experiment scripts, (2) assisting with the implementation of baselines and plotting scripts for visualizing results, (3) performing grammar and spelling checks to improve readability, and (4) proofreading the manuscript for clarity and correctness. All content, including the final text, figures, and scientific contributions, were curated and verified by the authors.

C Implementation Details

Our experiments are conducted using a framework based on the SGLang inference engine (Zheng et al., 2024). For all methods, we employ the inference engine to serve the Qwen2.5-7B-Instruct (Yang et al., 2024) model, which ensures efficient and consistent response generation across all experiments. We choose this model because of its popularity and moderate performance on evaluated benchmarks, leaving enough improvement space for TTS methods.

Calculation of REAR Scores The REAR score is calculated by obtaining token-level log-probabilities for each generated response under two distinct contexts: one with the full prompt including preference information and another with only the question part of the prompt. We use the SGLang frontend APIs to directly obtain the log-probabilities for each token in the response. The log-probabilities on the full prompt can be directly acquired within the text generation process, while the log-probabilities on the question part of the prompt are calculated with another simple forward process that takes the question part and the generated response as input, which can be lightweight and efficient. These two sets of log-probabilities are then combined as a weighted sum, controlled by the hyperparameter λ, to produce the final realignment score, as formulated in our methodology. To calculate the REAR score of a complete or partial response, we simply set the discount factor $\gamma = 1$ to take all the tokens into account with equal weights.

TTS Methods We adapt our REAR scores to two TTS methods, best-of-N sampling (BoN) (Stiennon et al., 2020) and dynamic verifier tree search (DVTS) (Beeching et al., 2025). For BoN, we directly use the inference engine to generate multiple responses in separate requests, and then select the response with the highest REAR score. For DVTS, we use the line break as the delimiter of each tree-search step, where the algorithm will select the expanded branch of each node according to the REAR score. In our experiments, unless specified, we set the number of samples to 16 for all BoN methods including the baselines. For DVTS, we set an equivalent compute budget to the BoN method by setting its expansion width and initial tree nodes both to 4. According to Beeching et al. (2025), this setting is comparable to the $N = 16$ setting for BoN. All the generated responses are sampled using a temperature of 1.0 and the maximum generated length is set to 2048 tokens.

Best-of-N with Generative RM (GenRM) This baseline leverages the base model as its own judge. Each generated response is appended with a template that prompts the model to evaluate whether the response is preferred. The final reward is calculated from the log-probability difference between the model generating “Yes” and “No”. To be specific, we use the following chat template:

Listing 1: Generative Verification Prompting Template

```
System: [Preference in the data sample]

User: [Question]

Assistant: [Response]

User: Please act as an impartial judge and evaluate the
    quality of the assistant’s response. A preferred response
    is helpful, harmless, and accurately follows instructions.
     Is this a preferred response? Answer ’Yes’ or ’No’ in the
     format ’Preferred: X’.
```

Table 2: Ablation study on the hyper-parameter λ in REAR on different tasks from PrefEval and Multifaceted benchmarks.

	λ			
Method	3	10	20	50
PrefEval Explicit Preference				
BoN w/ REAR	71.9	72.3	**74.1**	71.4
DVTS w/ REAR	**77.4**	76.4	76.3	75.1
PrefEval Implicit Choice				
BoN w/ REAR	76.8	77.2	**78.2**	77.4
DVTS w/ REAR	73.8	76.2	**78.6**	77.4
PrefEval Implicit Preference				
BoN w/ REAR	14.6	15.1	15.4	**16.2**
DVTS w/ REAR	14.7	17.4	**19.1**	18.1
Multifaceted Bench				
BoN w/ REAR	75.4	76.0	**76.3**	75.3
DVTS w/ REAR	74.5	75.3	**76.8**	75.6

```
Assistant: [Potential chain-of-thought reasoning process]
   Preferred: [Yes/No]
```

Best-of-N with External RM This approach uses an external, dedicated reward model, Skywork-Reward-Llama-8B (Liu et al., 2024a), hosted on an independent inference endpoint. For each candidate response, the prompt and the response are sent to this external model, which returns a scalar reward score.

Amulet and Linear Alignment (LA) We use the implementation of Amulet and LA provided by the Amulet paper (Zhang et al., 2025c) to run the experiments[1]. We do not change the default hyper-parameters of these baselines. For Amulet, experiments are run with an iteration number of 60 for test-time alignment.

D Evaluation Protocols

In this section, we provide a detailed description of the evaluation protocols used for each benchmark in our experiments. Except for the PrefEval implicit choice task, which uses the accuracy on selected option as the metric, the other tasks typically adopt LLM-as-a-judge for evaluation. We choose the GPT-4.1 model as the judge by calling the OpenAI API.

PrefEval The PrefEval benchmark (Zhao et al., 2025) is evaluated across its three distinct data types, each with a specific protocol. For **explicit preference**, the task is evaluated using an LLM-as-a-judge. For each generated response, a series of automated checks assesses different aspects of quality and preference alignment, including helpfulness, preference violation, consistency, and hallucination. The final score is an aggregated metric that reflects overall preference-following accuracy. The evaluation protocol for **implicit preference** is identical to that of explicit preference, using the same LLM-as-a-judge and the same set of automated checks. For **implicit choice**, this is a multiple-choice task where the model must select the best response from four options. The evaluation protocol extracts the model's choice from its generated output and compares it to the ground-truth correct answer. The final performance is measured by accuracy.

Multifaceted Bench For the Multifaceted Bench (Lee et al., 2024), we also employ an LLM-as-a-judge for evaluation. The judge assesses the model's generated response based on a set of rubrics

[1] https://github.com/zowiezhang/Amulet

Table 3: Ablation study on λ for REAR on Ping-Pong Bench.

BoN w/ REAR				
λ	3	10	20	50
Score	2.92	2.97	3.02	**3.07**
DVTS w/ REAR				
λ	0.3	0.5	1	1.5
Score	2.88	2.99	**3.03**	2.87

that are provided within each data sample. It assigns a score from 1 to 5 for each rubric. The final reported score is the average of these scores across all rubrics.

Ping-Pong Bench The Ping-Pong benchmark (Gusev, 2024) for role-playing is evaluated using an LLM-as-a-judge. The judge evaluates the entire conversation based on three main criteria: **stay-in-character score** (how well the model maintains its assigned persona), **entertaining score** (how engaging and entertaining the conversation is), and **fluency score** (the quality and naturalness of the language used). Each criterion is scored on a scale, and the final metric is the overall average score across these dimensions. We adopt the English version of the Ping-Pong-v2 dataset for evaluation. Differing from the original benchmark that uses gpt-4o-mini as the interrogator model to generate multi-turn data from the user side, we use the same model as our base model, i.e., Qwen2.5-7B-Instruct, as the interrogator model to avoid heavy expenses on calling the API. We note that this setting will result in slight performance degradation compared to the original benchmark. However, it is still able to capture the role-playing capabilities of the model and the comparisons are fair and valid for all evaluated methods.

E Additional Experiments on Hyper-parameter Tuning

We conduct an ablation study on the hyper-parameter λ in REAR, which controls the weight of the value function. The results are shown in Table 2 and Table 3. The results largely confirm the observations made in Section 5.2. Across most tasks on the PrefEval and Multifaceted benchmarks, we observe a non-monotonic relationship between λ and performance. For the majority of these tasks, the optimal performance is achieved when λ is around 20 for both BoN and DVTS. This reinforces the idea that there is a trade-off between adhering to user preference and maintaining the general quality of the response, as an excessively high λ can degrade helpfulness.

However, we also note some task-specific variations. For instance, on the PrefEval Explicit Preference task, DVTS achieves its best performance with a smaller λ of 3. On the PrefEval Implicit Preference and the Ping-Pong Bench tasks, BoN with REAR shows a trend of continuously improving performance as λ increases up to 50. This suggests that for certain tasks, particularly those requiring strong adherence to a persona (Ping-Pong) or subtle preference cues, a stronger emphasis on the preference-related reward component is beneficial.

Furthermore, the optimal range for λ appears to depend on the specific TTS algorithm. For example, the DVTS algorithm adopts a step-by-step tree search strategy, which can be more sensitive to the preference reward. Exaggerating the preference reward may lead to suboptimal performance. In contrast, BoN methods only rate the final response after finishing generation, where a large λ value is often preferred for the benchmark. For the Ping-Pong benchmark, DVTS achieves its peak performance at $\lambda = 1.0$, while BoN performs best with a much larger λ. This highlights that the interaction between the search strategy and the reward scaling is an important factor. In summary, while a λ of 20 serves as a robust default for many scenarios, fine-tuning this hyper-parameter for the specific task and TTS method can unlock further performance gains.