DIFFPO: Diffusion-styled Preference Optimization for Efficient Inference-Time Alignment of Large Language Models

Anonymous ACL submission

Abstract

Inference-time alignment provides an efficient alternative for aligning LLMs with humans. However, these approaches still face challenges, such as limited scalability due to policy-specific value functions and latency during the inference phase. In this paper, we propose a novel approach, Diffusion-styled Preference Optimization (**DIFFPO**), which provides an efficient and policy-agnostic solution for aligning LLMs with humans. By directly performing alignment at sentence level, **DIFFPO** avoids the time latency associated with token-level generation. Designed as a plug-and-play module, **DIFFPO** can be seamlessly integrated with various base models to enhance their alignment. Extensive experiments on AlpacaEval 2, MT-bench, and HH-RLHF demonstrate that **DIFFPO** achieves superior alignment performance across various settings, achieving a favorable trade-off between alignment quality and inference-time latency. Furthermore, **DIFFPO** demonstrates model-agnostic scalability, significantly improving the performance of large models such as Llama-3-70B.

1 Introduction

The alignment of large language models (LLMs) with human preferences has recently emerged as a focal area of research (Wang et al., 2023; Shen et al., 2023). Prominent techniques such as Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022) and Direct Preference Optimization (DPO) (Rafailov et al., 2024) have demonstrated substantial efficacy. However, these methods require the optimization of individual policies, posing challenges such as high consumption of training resources. Inference-time alignment (Mudgal et al., 2023; Han et al., 2024) provides an efficient alternative through direct adjustment of the model's output distribution, thus avoiding the need for resource-intensive retraining. Despite its advantages, this approach still requires policy-specific value functions, limiting its scalability across different models. Additionally, the inference-time latency remains high, presenting further challenges to its practical deployment.

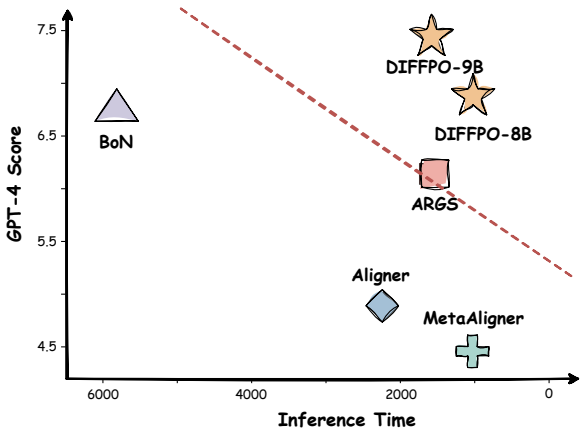

Figure 1: **Comparison with Inference-Time Methods.** Points closer to the ***top-right*** indicate a superior trade-off between performance and inference time.

In this paper, we investigate an efficient and policy-agnostic preference optimization method. We begin by reconsidering the objective of *aligning* with humans (Yao et al., 2023; Shen et al., 2023). As illustrated in Fig. 2(a), the alignment process operates at the sentence level, focusing on adjusting key components of the generated content, such as style or format, to better reflect human intentions or values. Inspired by the global controllability of the diffusion process (Li et al., 2022; Lyu et al., 2023), we propose Diffusion-styled Preference Optimization (**DIFFPO**). **DIFFPO** draws an analogy from the diffusion-based denoising process to model the iterative adjustment required for aligning human preferences, as shown in Fig. 2(b). By employing parallel decoding (Santilli et al., 2023; Leviathan et al., 2023), **DIFFPO** directly predicts sentence-level transitions, thus avoiding the time latency associated with token-level generation. During the training phase, we optimize the **DIFFPO** with an objective that maps generations with varying align-

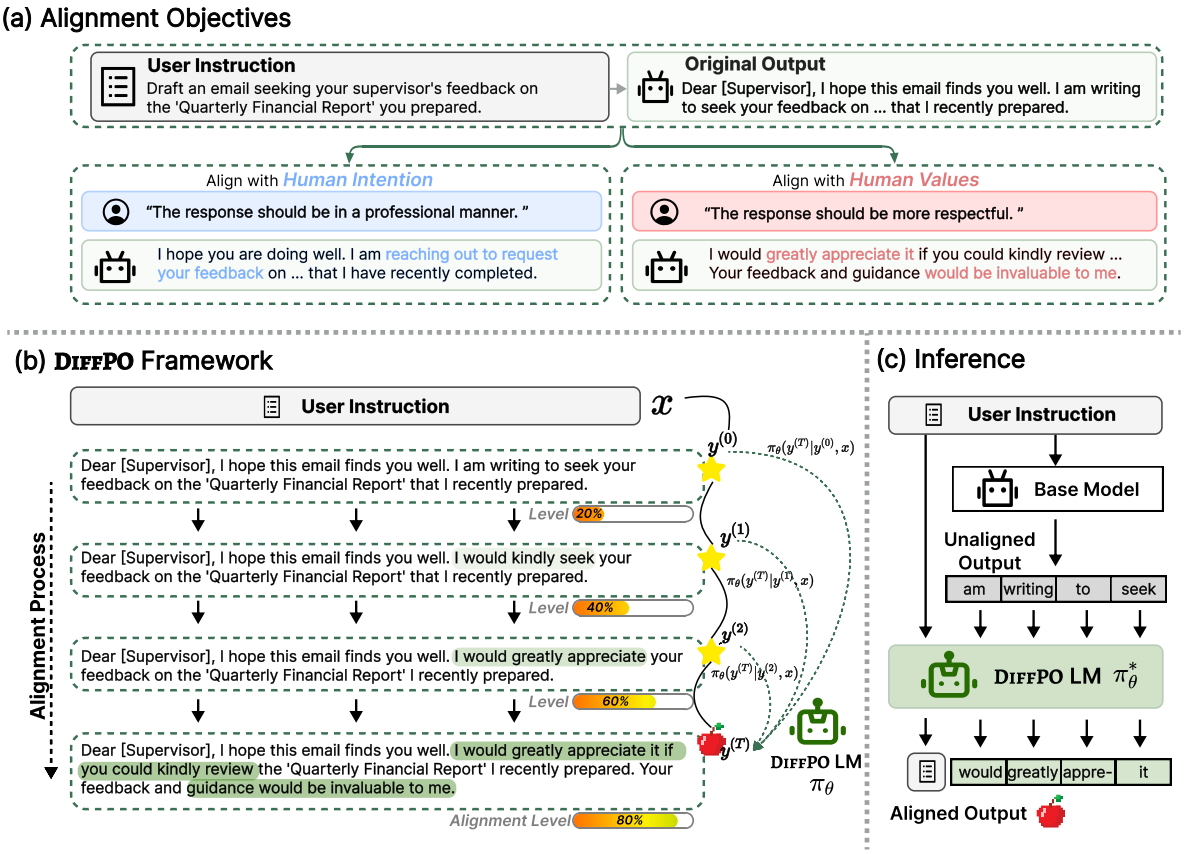

Figure 2: **Illustration of the DiffPO Framework.** (a) The objective of LLM alignment is to adjust the output of LLMs to reflect human values and intentions. In this process, preferences are considered at the **sentence level**, focusing on aspects such as the style and format of the complete output. (b) We propose Diffusion-style Preference Optimization (**DiffPO**), which reconceptualizes the alignment process as a sentence-level denoising process, where the goal is to transform an unaligned sentence $\mathbf{y}^{(0)}$ into an aligned sentence $\mathbf{y}^{(T)}$ step by step. (c) Designed as a plug-and-play module, **DiffPO** can be directly integrated with the base model output and yield better alignment.

ment levels to an aligned target, making it a policy-agnostic, plug-and-play module. The optimized **DiffPO** can then be seamlessly integrated with the output of the base model, enhancing its alignment level, as demonstrated in Fig. 2(c).

We evaluate the performance of **DiffPO** on several benchmark datasets, including AlpacaEval 2 (Dubois et al., 2024), MT-bench (Zheng et al., 2023), and HH-RLHF (Bai et al., 2022). Empirical results demonstrate that **DiffPO** achieves superior alignment performance across various base models and settings. Compared to inference-time alignment techniques, **DiffPO** strikes an optimal trade-off between alignment performance and inference-time latency, as shown in Fig. 1. Additional experiments highlight the model-agnostic scalability of **DiffPO** across different base models. Specifically, **DiffPO**-9B significantly enhances the performance of models such as Llama-3-70B and GPT-4o, showcasing its capability to improve weak-to-strong supervision.

The advantages of **DiffPO** can be summarized as:

- **Model-agnostic.** **DiffPO** is optimized to learn sentence-level refinement, independent of the specific base LLMs. This allows it to be applied across a variety of base LLMs. Furthermore, **DiffPO** does not require access to model parameters, which enhances its compatibility with API-based models and existing preference-aligned models.
- **Training and Inference Efficiency.** As a post-inference alignment strategy, **DiffPO** adopts a one-for-all approach: it involves training *one* single **DiffPO** and applying it *for all* base models, thus significantly reducing the resource intensiveness associated with policy optimization. Moreover, by framing alignment as sentence-level prediction, **DiffPO** bypasses the time latency associated with token-level generation, thereby improving inference-time efficiency.

2 Method

2.1 Preliminaries: Large Language Models

Next-Token Prediction. The text generation of autoregressive large language models (LLMs) with prompt $\mathbf{x}$ and response $\mathbf{y}$ can be modelled as a next-token prediction process. Given the input $\mathbf{x}$, The language model $\pi(\cdot|\mathbf{x})$ autoregressively maps from current tokens $(\mathbf{x}, \mathbf{y}_{1:n-1})$ to a distribution over the next token $\mathbf{y}_n$. The maximum token, N, sets the length limit for LLM outputs, which conclude with an end-of-sentence (EoS) token $\mathbf{y}_N = \text{EoS}$ that ends the generation. The generated output $\mathbf{y}$ consists of predicted tokens $(\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_N)$.

Alignment of LLMs. During the alignment of LLMs, the objective is to optimize a language model π_θ that maximizes the user's preference (Christiano et al., 2017; Ouyang et al., 2022; Rafailov et al., 2024):

$$\max_{\pi_\theta} \mathbb{E}_{\substack{x\sim D, \mathbf{y}\sim\pi_\theta(\mathbf{y}|\mathbf{x}) \\ \mathbf{y}'\sim\pi_{\text{ref}}(\mathbf{y}|\mathbf{x})}} [p(\mathbf{y} \succ \mathbf{y}'|\mathbf{x}) - \beta D_{KL}(\pi_\theta \| \pi_{\text{ref}})], \quad (1)$$

where $p(\mathbf{y} \succ \mathbf{y}'|\mathbf{x})$ represents the preference, i.e., the probability that $\mathbf{y}$ is preferred over $\mathbf{y}'$ given the context $\mathbf{x}$, which can be generally represented by the reward function r. The parameter β controls the deviation from the reference policy π_{ref}, which generally corresponds to the SFT model.

Parallel Decoding of LLMs. In comparison to next-token prediction, where token-level generation is performed sequentially to obtain a sentence, parallel decoding has demonstrated the capacity by enabling sentence-level generation and improving content quality (Santilli et al., 2023; Leviathan et al., 2023). Concretely, supposing

$$f(\mathbf{y}_n, \mathbf{y}_{<n}, \mathbf{x}) := \mathbf{y}_n - \arg\max_{\mathbf{y}} \pi(\mathbf{y}|\mathbf{y}_{<n}, \mathbf{x}),$$

parallel decoding re-frames the LLM inference process as solving a system of nonlinear equations w.r.t. all tokens in a sentence $\mathbf{y}_n$ for $n = 1, \ldots, N$. It can be solved in a parallel and iterative way:

$$\begin{cases} \mathbf{y}_1^{(t+1)} = \arg\max_{\mathbf{y}} \pi(\mathbf{y} \mid \mathbf{x}) \\ \mathbf{y}_2^{(t+1)} = \arg\max_{\mathbf{y}} \pi(\mathbf{y} \mid \mathbf{y}_1^{(t)}, \mathbf{x}) \\ \quad\vdots \\ \mathbf{y}_N^{(t+1)} = \arg\max_{\mathbf{y}} \pi(\mathbf{y} \mid \mathbf{y}_{<N}^{(t)}, \mathbf{x}) \end{cases} \quad (2)$$

In this way, for one forward pass of the LLM at time t, we can obtain the next sentence $\mathbf{y}^{(t+1)}$ based on the previous one $\mathbf{y}^{(t)}$.

2.2 Diffusion-styled Preference Optimization

Motivation. The goal of LLM alignment is to align the outputs of LLMs with human values or intentions (Yao et al., 2023). In this process, preferences are defined at the **sentence-level**, focusing on the style or format of complete generated answers, as illustrated in Fig. 2(a). However, the generation of these responses occurs at the token level, following the next-token prediction pattern inherent in LLM modeling. This requires existing alignment techniques to optimize preferences (or rewards) at the **token-level**, which complicates the learning process (Andrychowicz et al., 2017; Zhong et al., 2024; Zeng et al., 2024). This inconsistency prompts us to reconsider the formulation of the alignment process.

Reformulation. Inspired by the potential benefits of the diffusion process in controllable text generation (Gong et al., 2022; Han et al., 2022; Ye et al., 2024b), we draw an analogy between the aligning LLMs and the diffusion process. Specifically, we propose **Diffusion-styled Preference Optimization** (**DiffPO**), which reconceptualizes alignment as a sentence-level denoising process. The denoising process π gradually refines the initial unaligned output $\mathbf{y}^{(0)}$ by adjusting the format or style as a whole. This process ultimately produces the aligned output $\mathbf{y}^{(T)}$, as illustrated in Fig. 2(b). The sentence-level alignment process can be formulated as follows:

$$\pi(\mathbf{y}^{(0:T)}) := p(\mathbf{y}^{(0)}) \prod_{t=1}^{T} \pi(\mathbf{y}^{(t)}|\mathbf{y}^{(t-1)}, \mathbf{x}), \quad (3)$$

where $\mathbf{y}^{(0)}$ and $\mathbf{y}^{(T)}$ represent the initial unaligned and final aligned generations, respectively. The intermediate sequence $\mathbf{y}^{(1:T-1)}$ can be viewed as the unaligned generations progressively transitioning along the trajectory from $\mathbf{y}^{(0)}$ to $\mathbf{y}^{(T)}$.

Assuming the existence of a reward model $r(\mathbf{x}, \mathbf{y})$, which captures how well the generated output $\mathbf{y}$ aligns with human preferences given the input $\mathbf{x}$, the goal is to optimize a **DiffPO** model π_θ. This model learns to take a sentence as input and predict the next sentence with a higher reward, as illustrated in Fig. 2(c). The goal can be

expressed as follows:

$$\pi_\theta(\mathbf{y}^{(t)}|\mathbf{y}^{(t-1)},\mathbf{x}) \propto p(\mathbf{y}^{(t-1)},\mathbf{x})\exp(r(\mathbf{x},\mathbf{y}^{(t)})).$$

By employing parallel decoding, the **DiffPO** model directly performs sentence-level predictions.

2.3 Consistency Optimization of DiffPO

Inspired by Consistency LLMs (Kou et al., 2024), we propose to consistently map any intermediate (unaligned) generation $\mathbf{y}^{(t)}$ to the aligned generation $\mathbf{y}^{(T)}$. We jointly optimize the **DiffPO** model π_θ with two losses: one aligns the intermediate generation with the aligned generation, and the other prevents the corruption of the autoregressive (AR) modeling in the base model, thereby maintaining the generation quality.

Consistency Loss. For a prompt $\mathbf{x}$ with an unaligned generation $\mathbf{y}^{(t)}$, we directly guide the model to output $\mathbf{y}^{(T)}$ with $\mathbf{y}^{(t)}$ as the input by minimizing the following loss L_{Con}=

$$\mathbb{E}_{(\mathbf{x},\mathbf{y}^{(t)},\mathbf{y}^{(T)})\sim\mathcal{D}}\left[\sum_{i=1}^{N}\mathrm{KL}(\pi_{\theta^-}(\mathbf{y}^{(T)}_{<i},\mathbf{x})\|\pi_\theta(\mathbf{y}^{(t)}_{<i},\mathbf{x}))\right] \tag{4}$$

where $\theta^- = \mathrm{stopgrad}(\theta)$ and N denotes the length of generation. $\mathrm{KL}(\cdot\|\cdot)$ denotes the forward KL distance between two distributions.

AR Loss. To prevent the corruption of the autoregressive (AR) modeling in the base model and maintain the generation quality, we incorporate the AR loss based on the generated sequence $\mathbf{y}^{(T)}$:

$$L_{\text{AR}} = \mathbb{E}_{(\mathbf{x},\mathbf{y}^{(T)})\sim\mathcal{D}}\left[-\sum_{i=1}^{N}\log\pi_\theta(\mathbf{y}^{(T)}_i|\mathbf{y}^{(T)}_{<i},\mathbf{x})\right]. \tag{5}$$

The total loss with weight ω is:

$$L(\theta) = L_{\text{AR}} + \omega L_{\text{Con}}. \tag{6}$$

2.4 The Objective of DiffPO within RLHF

In this section, we analyze the role of **DiffPO** in achieving the goal of RLHF. We start with the same RL objective as prior work, Eq. 1, under a general reward function r^*. Following prior work (Peng et al., 2019; Rafailov et al., 2024), the optimal solution to the KL-constrained reward maximization objective in Eq. 1 takes the form: $r^*(\mathbf{x},\mathbf{y}) = \beta\log\left(\frac{\pi^*(\mathbf{y}|\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}|\mathbf{x})}\right) + \beta\log Z(\mathbf{x})$, where $Z(\mathbf{x}) = \sum_{\mathbf{y}}\pi_{\text{ref}}(\mathbf{y}|\mathbf{x})\exp\left(\frac{1}{\beta}r^*(\mathbf{x},\mathbf{y})\right)$ is the partition function. With Bradley-Terry model, we can represent the preference function as the difference of rewards for a preferred answer $\mathbf{y}_w$ and a dispreferred answer $\mathbf{y}_l$:

$$\begin{aligned} p(\mathbf{y}_w \succ \mathbf{y}_l|\mathbf{x}) &= \sigma(r^*(\mathbf{x},\mathbf{y}_w) - r^*(\mathbf{x},\mathbf{y}_l)) \\ &= \sigma\left(\beta\log\frac{\pi^*(\mathbf{y}_w \mid \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_w \mid \mathbf{x})} - \beta\log\frac{\pi^*(\mathbf{y}_l \mid \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_l \mid \mathbf{x})}\right). \end{aligned}$$

Substitute by $\pi^*(\mathbf{y} \mid \mathbf{x}) = \pi_{\textbf{DiffPO}}(\mathbf{y} \mid \mathbf{y}', x)$ $\pi_{\text{ref}}(\mathbf{y}' \mid \mathbf{x})$, we obtain $p(\mathbf{y}_w \succ \mathbf{y}_l|\mathbf{x})$ equals to

$$\sigma\left(\beta\log\frac{\pi_{\textbf{DiffPO}}(\mathbf{y}_w \mid \mathbf{y}_l,\mathbf{x})}{\pi_{\textbf{DiffPO}}(\mathbf{y}_l \mid \mathbf{y}_l,\mathbf{x})} - \beta\log\frac{\pi_{\text{ref}}(\mathbf{y}_w \mid \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_l \mid \mathbf{x})}\right). \tag{7}$$

Note that the first term in Eq. 7 is optimized through the consistency loss in Eq. 4 by maximizing the probability of predicting $\mathbf{y}_w$. The second term depends only on $\mathbf{x}$, with π_{ref} remaining constant. Moreover, the deviation from the base policy can be easily controlled, since $\mathbf{y}_w$ is derived from $\mathbf{y}_l$.

In summary, the objective of **DiffPO** *as defined in Eq. 6 aligns with the RLHF objective in Eq. 1.* Furthermore, since $\pi_{\textbf{DiffPO}}$ is optimized independently from the base model π_{ref}, it can be deployed in a model-agnostic manner.

2.5 Practical Implementations

Generate Alignment Trajectories. To implement **DiffPO**, we collect the alignment trajectory for each prompt, thereby forming an original training set $\mathcal{D}$. Specifically, for each prompt $\mathbf{x}$ from the *UltraFeedback* dataset (Cui et al., 2023), we generate T responses using different base models. We then employ *ArmoRM* (Wang et al., 2024) reward model to score these responses. The response with the highest score is selected as $\mathbf{y}^{(T)}$. The remaining five responses are ranked based on their scores to form $\mathbf{y}^{(0:T-1)}$. T is set to 6.

Training and Inference. During the training phase, we initialize our aligning model π_θ using three backbones of varying sizes: Gemma-2-it-2B/9B, and Llama-3-8B-Instruct. The **DiffPO** model is optimized adhering to the optimization loss in Eq. 6 with parameters $N = 256$ and $w = 10^3$. Given the variable lengths of generations in $\mathcal{D}$, we standardize their lengths through padding or truncation. In the inference phase, the optimized model π^*_θ is employed to align responses from the vanilla generations produced by base models. Appendix B.1 shows more implementation details.

3 Experiment

3.1 Experiment Setup

Evaluation Benchmarks and Metrics. We conduct our experiments using two widely recognized benchmarks for open-ended instruction-following: *MT-Bench* (Zheng et al., 2023) and *AlpacaEval 2* (Dubois et al., 2024). These benchmarks are designed to evaluate the conversational abilities of models across a diverse set of queries. *AlpacaEval 2* includes 805 questions drawn from five distinct datasets, while *MT-Bench* covers eight categories and comprises a total of 80 questions. Additionally, we employ the *HH-RLHF* (Bai et al., 2022) datasets to assess how well the models' generative capabilities align with human values, particularly emphasizing helpfulness and harmlessness. We adhere to each benchmark's specific evaluation protocol to report scores. In *AlpacaEval 2*, we report both the raw win rate (WR) and the length-controlled win rate (LC), comparing performance against the GPT-4 model. In contrast, we present the average score for *MT-Bench*, also utilizing GPT-4 as the judge model. For *HH-RLHF*, we report scores that reflect the models' helpfulness and harmlessness, as well as the overall score. These scores are measured using *ArmoRM* (Wang et al., 2024), a state-of-the-art reward model from *RewardBench* (Lambert et al., 2024), designed to align with human preferences.

Baselines. We compare **DiffPO** with two primary categories of offline preference optimization methods. In the category of **training-based** methods: *Direct Preference Optimization (DPO)* (Rafailov et al., 2024) reparameterizes reward functions to simplify and stabilize the preference learning process. *SimPO* (Meng et al., 2024) utilizes the average log probability of a sequence as an implicit reward, aligning more closely with model generation. For **training-free** methods: *Black-Box Prompt Optimization (BPO)* (Cheng et al., 2024) adapts user prompts to better align with LLMs' input comprehension, achieving user intents optimally without altering LLM parameters. *ARGS* (Khanov et al., 2024) integrates alignment into the decoding process through reward-guided search, eliminating the need for costly RL training. *Best-of-N sampling (BoN)* (Nakano et al., 2021) samples N times and selects the highest-scoring sample based on the reward model, with N set to 4 in our experiments using *ArmoRM* (Wang et al., 2024) as the reward model. Furthermore, *Aligner* (Ji et al., 2024) and *MetaAligner* (Yang et al., 2024a) employ an additional model to learn corrective residuals between preferred and dispreferred responses to refine model generation.

Base Models and Inference Settings. We perform preference optimization primarily on two model families: *Llama-3-8B* (AI@Meta, 2024) and *Mistral-7B* (Jiang et al., 2023), under two configurations: SFT and Instruct. In the SFT configuration, we utilize open-source models from *SimPO* (Meng et al., 2024) that follow *Zephyr* (Tunstall et al., 2023) to train the base models (i.e., `meta-llama/Meta-Llama-3-8B`) on the *UltraChat-200k* (Ding et al., 2023) dataset to derive an SFT model. For the Instruct configuration, we employ an off-the-shelf instruction-tuned models (i.e., `meta-llama/Meta-Llama-3-8B-Instruct`). To further validate scalability, we conduct additional experiments using the *Llama-3.2* series, *Qwen-2.5* series (Team, 2024), and GPT-4o (Achiam et al., 2023) as the base models.

During the inference phase of **DiffPO**, we initially generate responses using the base models. For each benchmark. In *AlpacaEval 2* and *HH-RLHF*, we employ a sampling decoding strategy with a temperature setting of 0.7. For *MT-Bench*, we adhere to the official decoding configuration, which specifies varying temperatures for different categories. In our primary experiments, we set the maximum token generation length to 256. Results for experiments conducted at various lengths are provided in Tab. 4. Subsequently, the responses generated by the base models are aligned using the trained **DiffPO**. For the main results, parallel decoding is executed with a block size of 256.

3.2 Experiment Results

DiffPO significantly outperforms existing preference optimization methods. As shown in Table 1, while all preference optimization algorithms improve performance over the base model, **DiffPO** achieves the best overall performance across all benchmarks and settings. These consistent and significant improvements underscore the robustness and effectiveness of **DiffPO**. Notably, **DiffPO** outperforms the training-based baselines (i.e., SimPO and DPO) across various settings, despite requiring only a single training session of **DiffPO** model and being capable of enhancing the performance of multiple base models.

Method	Llama-3-SFT (8B)					Llama-3-Instruct (8B)				
	MT-bench	AlpacaEval 2		HH-RLHF		MT-bench	AlpacaEval 2		HH-RLHF	
	GPT-4	LC (%)	WR (%)	Helpful	Harmless	GPT-4	LC (%)	WR (%)	Helpful	Harmless
Base Model	6.21	22.09	20.81	0.59	0.91	6.78	36.83	42.12	0.67	0.93
w. DPO	6.59	29.84	36.77	0.68	0.89	6.90	47.20	53.56	0.74	0.92
w. SimPO	6.62	32.27	40.96	0.66	0.86	7.05	52.57	58.33	**0.75**	0.92
w. BPO	5.84	21.34	22.33	0.60	0.92	6.43	22.39	34.06	0.67	0.92
w. ARGS	6.14	9.06	13.97	0.49	0.86	6.84	31.83	34.74	0.64	0.89
w. BoN	6.79	35.14	32.26	0.62	0.92	6.89	45.10	49.94	0.67	0.92
w. Aligner	4.88	20.41	17.15	0.60	0.91	4.82	32.53	32.69	0.67	0.96
w. MetaAligner	4.46	19.81	18.23	0.52	0.89	4.50	20.75	19.08	0.52	0.91
w. **DiffPO**-8B	6.96	36.24	40.96	0.62	0.93	7.02	36.44	41.01	0.68	0.93
w. **DiffPO**-9B	**7.45**	**49.72**	**54.23**	**0.71**	**0.98**	**7.40**	**55.84**	**61.88**	0.72	**0.98**
Method	**Mistral-SFT (7B)**					**Mistral-Instruct (7B)**				
	MT-bench	AlpacaEval 2		HH-RLHF		MT-bench	AlpacaEval 2		HH-RLHF	
	GPT-4	LC (%)	WR (%)	Helpful	Harmless	GPT-4	LC (%)	WR (%)	Helpful	Harmless
Base Model	5.73	20.15	17.24	0.56	0.87	6.39	32.81	34.86	0.66	0.94
w. DPO	5.91	31.28	32.65	0.66	0.91	6.29	35.60	37.73	0.67	0.92
w. SimPO	6.17	31.16	33.72	0.63	0.86	6.36	35.78	40.21	0.67	0.93
w. BPO	5.55	18.23	17.23	0.64	0.92	5.99	19.61	27.49	0.66	0.93
w. ARGS	5.12	11.07	13.95	0.55	0.87	6.20	26.60	29.68	0.66	0.92
w. BoN	6.21	33.36	27.74	0.64	0.94	6.40	34.75	39.24	0.68	0.94
w. Aligner	4.27	18.27	15.53	0.60	0.95	4.42	28.88	30.30	0.66	0.93
w. MetaAligner	4.08	12.40	9.72	0.51	0.85	3.71	18.55	16.91	0.55	0.91
w. **DiffPO**-8B	6.87	34.42	40.08	0.62	0.88	7.04	35.92	40.70	0.68	0.92
w. **DiffPO**-9B	**7.13**	**48.99**	**52.87**	**0.70**	**0.96**	**7.33**	**56.22**	**61.71**	**0.72**	**0.98**

Table 1: **Comparison results with baseline methods. DiffPO** achieves the superior alignment performance across all benchmarks, outperforming the training-based baselines (i.e., SimPO and DPO) in various settings. Notably, **DiffPO** requires only a single training session and is applicable to multiple base models. The best result is highlighted in **bold**, while the second-best result is highlighted with underline.

DiffPO consistently improves the performance of base models of various sizes. We report the performance of **DiffPO**-2B and **DiffPO**-9B on base models of various sizes, with the results presented in Table 2. The results demonstrate that both **DiffPO**-2B and **DiffPO**-9B lead to performance improvements across different base models. However, the performance gain of **DiffPO**-2B is limited, showing notable improvements primarily for smaller models. In contrast, **DiffPO**-9B enhances the performance of larger models, such as Qwen2.5-14B and 32B, as well as black-box GPT-4, exhibiting a weak-to-strong improvement pattern. Furthermore, the results show that **DiffPO** can be effectively integrated with existing preference optimization methods, such as DPO and SimPO, further enhancing alignment performance. These results underscore the scalability of **DiffPO**.

DiffPO achieves a surpassing performance-efficiency trade-off. We compare **DiffPO** with existing inference-time alignment techniques, evaluating both alignment performance and execution time. The results are illustrated in Fig. 3, with the execution time measured on a single NVIDIA A100 80GB GPU. Points located closer to the top-right corner indicate a more favorable Pareto frontier. BoN and MetaAligner achieves commendable alignment performance and inference time respectively. However, when considering both aspects, **DiffPO** demonstrates a surpassing performance-efficiency trade-off on all three datasets. The experiments are conducted on Llama-3-SFT.

3.3 Analysis

Performance Under Hybrid Decoding. We investigate the hybrid decoding strategy of **DiffPO**, with results provided in Tab 3. We segment the vanilla generation, which has a maximum length of 256, into blocks of varying sizes and sequentially apply **DiffPO**-8B to each block. This approach allows **DiffPO** decoding to be parallel within blocks and auto-regressive between blocks. It can be observed that hybrid decoding significantly reduces the decoding time, with optimal efficiency achieved at a block size of 32. On the other

Base Models	MT-bench			AlpacaEval 2		HH-RLHF		
	1-Turn	2-Turn	Avg.	LC (%)	WR (%)	Overall	Helpful	Harmless
Llama-3.2-1B-Instruct	5.32	5.13	5.25	15.57	19.09	0.0955	0.5978	0.9313
w. **DiffPO**-2B	6.97	5.97	6.47	39.42	44.20	0.1077	0.6948	0.9728
w. **DiffPO**-9B	7.56	6.95	7.30	50.70	56.08	0.1130	0.7059	0.9770
Llama-3.2-3B-Instruct	6.84	6.06	6.45	33.41	37.43	0.1037	0.6533	0.9183
w. **DiffPO**-2B	7.13	6.46	6.79	39.94	45.32	0.1069	0.6956	0.9682
w. **DiffPO**-9B	7.58	7.00	7.36	54.06	59.30	0.1132	0.7106	0.9875
Llama-3-8B-SFT+DPO	6.70	6.48	6.59	29.84	36.77	0.1044	0.6814	0.8900
w. **DiffPO**-9B	7.42	7.03	7.22	54.29	59.51	0.1134	0.7178	0.9765
Llama-3-8B-SFT+SimPO	6.63	6.61	6.62	32.27	40.96	0.1022	0.6640	0.8589
w. **DiffPO**-9B	7.59	7.08	7.42	55.66	60.67	0.1121	0.7156	0.9638
Llama-3-8B-it+DPO	6.75	7.05	6.90	47.20	53.56	0.1120	0.7387	0.9154
w. **DiffPO**-9B	7.79	6.98	7.39	58.56	63.60	0.1140	0.7211	0.9831
Llama-3-8B-it+SimPO	7.09	7.00	7.05	52.57	58.33	0.1143	0.7483	0.9182
w. **DiffPO**-9B	7.43	7.22	7.33	59.66	65.32	0.1142	0.7229	0.9756
Llama-3-70B-Instruct	7.41	7.59	7.5	46.14	51.12	0.1087	0.6928	0.9163
w. **DiffPO**-9B	8.23	7.28	7.75	58.18	62.34	0.1137	0.719	0.9757
Qwen2.5-3B-Instruct	6.73	5.59	6.16	35.52	40.42	0.1050	0.6802	0.9587
w. **DiffPO**-2B	7.06	6.26	6.66	42.63	47.83	0.1065	0.6973	0.9771
w. **DiffPO**-9B	7.58	7.24	7.41	55.71	61.43	0.1132	0.7106	0.9875
Qwen2.5-7B-Instruct	7.11	6.96	7.03	45.03	49.95	0.1095	0.6995	0.9442
w. **DiffPO**-2B	7.01	6.34	6.67	43.89	49.07	0.1074	0.7013	0.9659
w. **DiffPO**-9B	7.62	7.10	7.35	57.89	63.01	0.1117	0.7100	0.9445
Qwen2.5-14B-Instruct	7.24	6.71	6.98	51.60	57.10	0.1117	0.7100	0.9445
w. **DiffPO**-2B	7.08	6.33	6.71	43.70	48.76	0.1078	0.7017	0.9704
w. **DiffPO**-9B	7.62	7.35	7.48	55.13	60.65	0.1136	0.7185	0.9759
Qwen2.5-32B-Instruct	7.35	6.95	7.15	54.93	60.95	0.1132	0.7189	0.9594
w. **DiffPO**-9B	7.58	7.63	7.60	55.13	61.94	0.1143	0.7248	0.9797
GPT-4o (API)	7.40	7.47	7.43	53.64	62.01	0.1119	0.6974	0.9669
w. **DiffPO**-9B	7.66	7.37	7.51	58.91	64.30	0.1129	0.7167	0.9893

Table 2: **Performance of DiffPO models.** The results demonstrate that both **DiffPO**-2B and **DiffPO**-9B lead to performance improvements across different base models. **DiffPO**-9B enhances the performance of larger models, such as Qwen2.5-14B and 32B, as well as black-box GPT-4o, exhibiting a weak-to-strong improvement pattern. Furthermore, the results show that **DiffPO** can be effectively integrated with existing preference optimization methods, such as DPO and SimPO, further enhancing alignment performance.

hand, performance is enhanced when the block size is set to 256, which corresponds to purely parallel decoding, indicating a feasible trade-off. The experiments are conducted on Llama-3-SFT.

Scaling towards Longer Generation Lengths. We validate the scalability of the **DiffPO** model in response to increasing generation lengths, with results presented in Tab 4. Using base models, we generate outputs on *MT-Bench* under various maximum length settings and observe a positive correlation between increased text length and higher scores. Subsequently, the same optimized **DiffPO**-8B and 9B is applied to these outputs using the hybrid decoding strategy described in the previous section. This approach consistently yields enhanced alignment performance, demonstrating **DiffPO**'s robust scaling capabilities towards longer generation lengths.

Loss and Hyperparameter Ablation. We evaluate the effectiveness of the training loss of **DiffPO** in Section 2.3 and the inference strategy in Section 2.5. The results are presented in Table 5. We report on two decoding strategies: *vanilla decoding* of a single model and **DiffPO** *decoding*, which applies the optimized **DiffPO**-9B on the output of the base model. The findings indicate that applying **DiffPO** to the base model achieves performance superior to that of single

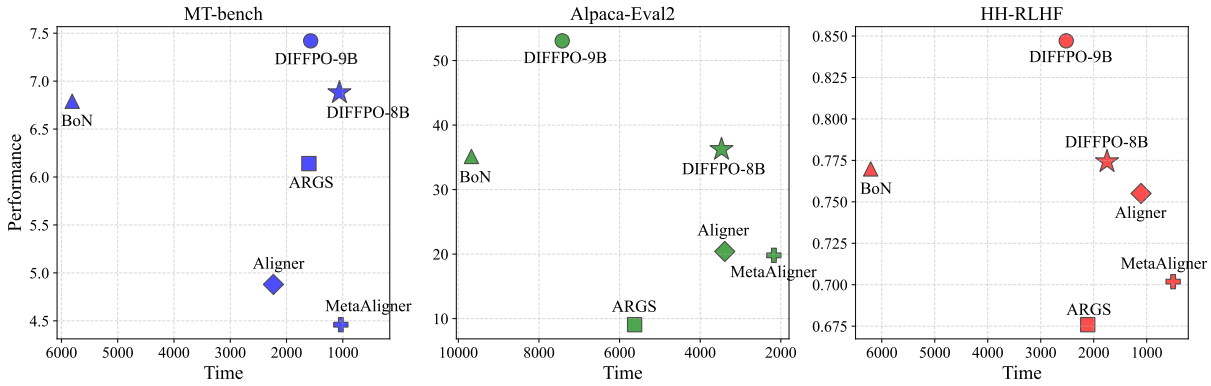

Figure 3: **Comparison of Inference-Time Efficiency.** We compare **DIFFPO** with existing inference-time alignment techniques, evaluating both alignment performance and execution time. Points located closer to the ***top-right*** corner indicate a better trade-off. When considering both aspects, **DIFFPO** demonstrates a surpassing performance-efficiency trade-off on all three datasets.

Block Size	**16**	**32**	**64**	**128**	**256**
MT (GPT-4)	6.86	**6.96**	6.84	6.81	6.77
Time (s)	1080	**1012**	1390	1520	1937
AE2 (LC)	33.52	33.54	33.46	32.98	**36.24**
Time (s)	3712	**3471**	4614	5510	7520
HH (Avg.)	0.7742	0.7749	0.7743	0.7741	**0.7761**
Time (s)	1564	**1551**	1620	1816	2684

Table 3: **Performance Under Hybrid Decoding.** We segment the vanilla generation into blocks of varying sizes and sequentially apply **DIFFPO**-8B to each block. This approach allows **DIFFPO** decoding to be parallel within blocks and auto-regressive between blocks. Hybrid decoding significantly reduces the decoding time, indicating a feasible trade-off for performance.

models alone, thus demonstrating the effectiveness of the **DIFFPO** strategy. Furthermore, we report the results of an ablation study on the hyperparameter w in Eq. 6. When using **DIFFPO** decoding, employing L_{Con} with larger values of w lead to a more pronounced improvement in performance.

4 Conclusion

This paper introduces a novel inference-time alignment framework for large language models, **DIFFPO**. **DIFFPO** achieves alignment at the sentence level to better model human preferences, drawing inspiration from the denoising process. **DIFFPO** outperforms both strong training-based and inference-time alignment techniques in terms of alignment performance and inference speed. Experiments scaling **DIFFPO** from 2B to 9B parameters, expanding the base model from 1B to 70B, and increasing the context length from 256 to 2,048 demonstrate that **DIFFPO** is a robust and scalable framework for LLM alignment.

Generation Length	256	512	1,024	2,048
Llama-3-SFT	6.21	6.61	6.76	6.71
w. **DIFFPO**-8B (Δ)	+0.75	+0.81	+0.93	+1.05
w. **DIFFPO**-9B (Δ)	+1.24	+1.64	+1.48	+0.50
Llama-3-Instruct	6.78	7.87	7.99	8.00
w. **DIFFPO**-8B (Δ)	+0.24	-0.12	+0.01	+0.02
w. **DIFFPO**-9B (Δ)	+0.62	+0.68	+0.34	+0.62
Mistral-SFT	5.73	6.42	6.50	6.36
w. **DIFFPO**-8B (Δ)	+1.14	+1.16	+1.25	+1.51
w. **DIFFPO**-9B (Δ)	+1.40	+1.47	+1.71	+1.86
Mistral-Instruct	6.39	7.47	7.68	7.64
w. **DIFFPO**-8B (Δ)	+0.65	+0.06	+0.13	+0.17
w. **DIFFPO**-9B (Δ)	+0.94	+0.59	+0.58	+0.78

Table 4: **Scaling towards Longer Generation Lengths.** We evaluate the performance of **DIFFPO** under various maximum length settings. When the same optimized **DIFFPO**-8B and 9B is applied to these outputs, consistently enhanced performance demonstrates **DIFFPO**'s robust scaling capabilities.

Loss	**MT**	**AE2**	**HH**
	GPT-4	**LC (%)**	**Avg.**
Vanilla Decoding			
Gemma-2-9B-it	7.37	54.15	0.8265
DIFFPO-9B	7.42	58.19	0.8279
Gemma-2-9B-it w. **DIFFPO**-9B			
L_{AR} only	7.41	58.01	0.8301
$10 \times L_{\text{Con}} + L_{\text{AR}}$	7.56	59.27	0.8389
$100 \times L_{\text{Con}} + L_{\text{AR}}$	7.55	59.43	0.8435
$1,000 \times L_{\text{Con}} + L_{\text{AR}}$	**7.60**	**60.56**	**0.8438**

Table 5: **Loss and Hyperparameter Ablation.** The results indicate that applying **DIFFPO**-9B to the base model (i.e., gemma-2-9B) yields outperforming performance than vanilla decoding of any single models, demonstrating the effectiveness of **DIFFPO** strategy.

Limitations

We acknowledge the presence of certain limitations. While **DiffPO** has demonstrated a superior trade-off between performance and inference-time cost, it still introduces additional inference latency due to the need for an extra model for alignment. Moreover, we observe that the performance of **DiffPO** scales with its size, which presents challenges for cost-effectiveness during deployment. Additionally, despite the empirical success and intuitive motivation behind **DiffPO**, a more rigorous theoretical analysis is required to fully understand its effectiveness. Future work could explore how to combine the diffusion process (i.e., the denoising process) with the alignment task more effectively. This paper draws insights from the analogy between the denoising process and alignment. We hope our findings will facilitate future exploration of existing successful techniques in the natural language processing domain.

Potential Risks

As an inference-time alignment technique, **DiffPO** aims to develop AI assistants that align with positive human intentions and social values. However, there is a potential risk that **DiffPO** could be misused to align with harmful or negative values. We strongly oppose any such misuse, as it could hinder human progress, and advocate for the responsible and ethical use of **DiffPO**.

References

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

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. 2024. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*.

AI@Meta. 2024. Llama 3 model card.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. 2017. Hindsight experience replay. *Advances in neural information processing systems*, 30.

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

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

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.

Jiale Cheng, Xiao Liu, Kehan Zheng, Pei Ke, Hongning Wang, Yuxiao Dong, Jie Tang, and Minlie Huang. 2024. Black-box prompt optimization: Aligning large language models without model training. *Preprint*, arXiv:2311.04155.

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

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback. *Preprint*, arXiv:2310.01377.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*.

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*.

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

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. 2024. Break the sequential dependency of llm inference using lookahead decoding. *arXiv preprint arXiv:2402.02057*.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and LingPeng Kong. 2022. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2023. Diffuseq-v2: Bridging discrete and continuous text spaces for accelerated seq2seq diffusion models. *arXiv preprint arXiv:2310.05793*.

Ishaan Gulrajani and Tatsunori B Hashimoto. 2024. Likelihood-based diffusion language models. *Advances in Neural Information Processing Systems*, 36.

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

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. 2022. Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. *arXiv preprint arXiv:2210.17432*.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11170–11189.

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

Jiaming Ji, Boyuan Chen, Hantao Lou, Donghai Hong, Borong Zhang, Xuehai Pan, Juntao Dai, Tianyi Qiu, and Yaodong Yang. 2024. Aligner: Efficient alignment by learning to correct. *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*.

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

Siqi Kou, Lanxiang Hu, Zhezhi He, Zhijie Deng, and Hao Zhang. 2024. Cllms: Consistency large language models. *arXiv preprint arXiv:2403.00835*.

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

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343.

Yuhui Li, Fangyun Wei, Jinjing Zhao, Chao Zhang, and Hongyang Zhang. 2023a. Rain: Your language models can align themselves without finetuning. *arXiv preprint arXiv:2309.07124*.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. 2023b. Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. In *Forty-first International Conference on Machine Learning*.

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*.

Justin Lovelace, Varsha Kishore, Chao Wan, Eliot Shekhtman, and Kilian Q Weinberger. 2024. Latent diffusion for language generation. *Advances in Neural Information Processing Systems*, 36.

Yiwei Lyu, Tiange Luo, Jiacheng Shi, Todd C Hollon, and Honglak Lee. 2023. Fine-grained text style transfer with diffusion-based language models. *arXiv preprint arXiv:2305.19512*.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. Simpo: Simple preference optimization with a reference-free reward. *Preprint*, arXiv:2405.14734.

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*.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

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.

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

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

Andrea Santilli, Silvio Severino, Emilian Postolache, Valentino Maiorca, Michele Mancusi, Riccardo Marin, and Emanuele Rodolà. 2023. Accelerating transformer inference for translation via parallel decoding. *arXiv preprint arXiv:2305.10427*.

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

Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. 2023. Large language model alignment: A survey. *arXiv preprint arXiv:2309.15025*.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. 2023. Consistency models. *arXiv preprint arXiv:2303.01469*.

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.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. 2023. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In *EMNLP*.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*.

Kailai Yang, Zhiwei Liu, Qianqian Xie, Jimin Huang, Tianlin Zhang, and Sophia Ananiadou. 2024a. Metaaligner: Towards generalizable multi-objective alignment of language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Shentao Yang, Shujian Zhang, Congying Xia, Yihao Feng, Caiming Xiong, and Mingyuan Zhou. 2024b. Preference-grounded token-level guidance for language model fine-tuning. *Advances in Neural Information Processing Systems*, 36.

Jing Yao, Xiaoyuan Yi, Xiting Wang, Jindong Wang, and Xing Xie. 2023. From instructions to intrinsic human values–a survey of alignment goals for big models. *arXiv preprint arXiv:2308.12014*.

Jiacheng Ye, Jiahui Gao, Shansan Gong, Lin Zheng, Xin Jiang, Zhenguo Li, and Lingpeng Kong. 2024a. Beyond autoregression: Discrete diffusion for complex reasoning and planning. *arXiv preprint arXiv:2410.14157*.

Jiacheng Ye, Shansan Gong, Liheng Chen, Lin Zheng, Jiahui Gao, Han Shi, Chuan Wu, Xin Jiang, Zhenguo Li, Wei Bi, et al. 2024b. Diffusion of thoughts: Chain-of-thought reasoning in diffusion language models. *arXiv preprint arXiv:2402.07754*.

Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. 2024. Token-level direct preference optimization. *arXiv preprint arXiv:2404.11999*.

Yizhe Zhang, Jiatao Gu, Zhuofeng Wu, Shuangfei Zhai, Joshua Susskind, and Navdeep Jaitly. 2024. Planner: generating diversified paragraph via latent language diffusion model. *Advances in Neural Information Processing Systems*, 36.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot

arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

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

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 Related Works

A.1 Align LLM with Human Preference.

A prominent approach to learning from human preferences is RLHF (Ouyang et al., 2022; Stiennon et al., 2020; Christiano et al., 2017; Bai et al., 2022). In this framework, a reward model is first trained, followed by the training of a bandit policy using Proximal Policy Optimization (PPO) (Schulman et al., 2017). Recent advancements such as direct preference optimization (DPO) (Rafailov et al., 2024; Meng et al., 2024; Ethayarajh et al., 2024) optimize the bandit policy directly from human preferences, bypassing the need for a reward model. These approaches are simpler to implement and require fewer computational resources. Inference-time approaches, on the other hand, achieve alignment by customizing the output of large language models (LLMs) during the decoding phase, without the need for parameter optimization. This results in enhanced flexibility and efficiency (Khanov et al., 2024; Mudgal et al., 2023). One representative method treats the text-generation process as a search problem, guided by external rewards (Huang et al., 2024; Han et al., 2024; Chakraborty et al., 2024). Another category of methods focuses on learning to refine the generated text (Li et al., 2023a; Ji et al., 2024; Yang et al., 2024a).

Token and Sentence-level. Existing training-based or inference-time alignment approaches typically rely on token-level rewards, while human preferences are generally provided and defined at the sentence level (Li et al., 2023b; Ahmadian et al., 2024; Zeng et al., 2024). To address this discrepancy, some works (Lightman et al., 2023; Yang et al., 2024b; Zeng et al., 2024) leverage token-wise or step-wise information to improve alignment performance. In contrast, this paper proposes modeling alignment as a sentence-level denoising process. We introduce a model-agnostic, inference-time alignment method, and our empirical results demonstrate its superiority in both performance and efficiency.

A.2 Parallel Decoding and Diffusion Process.

Parallel Decoding of LLMs Parallel decoding has been increasingly utilized and developed in recent research to accelerate the inference processes of large language models (LLMs). One line of research, including works by Leviathan et al. (2023); Chen et al. (2023), focuses on speculative decoding.

These techniques enhance LLM decoding speed by employing a smaller draft model to predict the outputs, which are then verified in parallel by a larger target model. Another research trajectory explores parallel decoding strategies that do not rely on a draft model. Methods such as conditioning on "look-ahead" tokens or employing Jacobi iterations have been investigated by Santilli et al. (2023); Fu et al. (2024). These approaches allow the target model to produce several tokens simultaneously, aiming for rapid convergence to a fixed point on a Jacobi trajectory. CLLMs (Song et al., 2023) develop a novel approach, refining the target LLM to consistently predict the fixed point from any given state.

Text Diffusion Models Diffusion models have demonstrated significant diversity and controllability in image generation (Ho et al., 2020; Song et al., 2020; Dhariwal and Nichol, 2021). Recently, these models have been extended to text generation, as evidenced by the works of (Li et al., 2022; Gong et al., 2022; Lovelace et al., 2024). In essence, diffusion models execute a multi-step denoising process that progressively transforms random noise into a coherent data sample. In the context of text, diffusion models can be considered an evolution of traditional iterative Non-Autoregressive models, as described by Gong et al. (2022). These models have demonstrated the ability to match or surpass Autoregressive (AR) models in terms of text perplexity (Han et al., 2022; Gulrajani and Hashimoto, 2024), diversity (Gong et al., 2023; Zhang et al., 2024), and various sequence-to-sequence tasks (Ye et al., 2024b,a).

Connection with DiffPO In this paper, we are motivated by the goal of aligning Large Language Models (LLMs) with human values or intentions, as outlined in (Yao et al., 2023). We define preferences at the sentence-level, focusing on the style or format of complete answers generated by the LLMs. If we consider each iteration of parallel decoding as a transition between states, this bears a formal resemblance to discrete diffusion models. In **DiffPO**, we leverage parallel decoding to implement sentence-level denoising, thereby enhancing the modeling of the alignment process.

The development of **DiffPO** is also inspired by Consistency Models (Song et al., 2023) and CLLMs (Kou et al., 2024). Consistency models address the limitation of the slow iterative sampling process by mapping any point along the probability flow ODE of the diffusion process back to the original point in a single step. CLLMs propose accelerating LLM inference by mapping the intermediate process of LLM parallel decoding to the final process. Similar to these works, we optimize **DiffPO** with consistency loss, thus enabling model-agnostic alignment.

A.3 Align LLM with Human Preference.

A prominent approach to addressing the challenge of learning from human preferences is reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022; Stiennon et al., 2020). The classic RLHF framework, initially proposed in Christiano et al. (2017) and Ziegler et al. (2019), has been further refined in Ouyang et al. (2022) and Bai et al. (2022). In this framework, a reward model is first trained, followed by the training of a bandit policy using Proximal Policy Optimization (PPO) (Schulman et al., 2017).

Recent advancements in direct preference optimization (DPO) (Rafailov et al., 2024) and related works such as Meng et al. (2024), Ethayarajh et al. (2024), and Hong et al. (2024), optimize the bandit policy directly from human preferences, bypassing the need for a reward model. These approaches are simpler to implement and require fewer computational resources.

Inference-time approaches, on the other hand, achieve alignment by customizing the output of large language models (LLMs) during the decoding phase, without the need for parameter optimization. This results in enhanced flexibility and efficiency (Khanov et al., 2024; Mudgal et al., 2023). One representative method treats the text-generation process as a search problem, guided by external rewards (Huang et al., 2024; Han et al., 2024; Chakraborty et al., 2024). Another category of methods focuses learning to refine the generated text (Li et al., 2023a; Ji et al., 2024; Yang et al., 2024a).

B Experiment

B.1 Experimental Setups

Training Details. As for the training set, we collect 6 generations from 6 base models (i.e., Llama-3-8B-Instruct, Llama-3-8B-SFT, Mistral-7B-SFT, Mistral-7B-Instruct, Gemma-2-2B-Instruct, Gemma-2-9B-Instruct). We then employ *ArmoRM* (Wang et al., 2024) to score these responses. The response with the highest score

Table 6: **Comparison results of DIFFPO models.** The experiments are conducted on base models of Qwen-2.5-7B and 14B. It shows that **DIFFPO** consistently achieves superior performance across various base models.

Base Models	MT-bench			AlpacaEval 2		HH-RLHF		
	1-Turn	2-Turn	Avg.	LC (%)	WR (%)	Overall	Helpful	Harmless
Qwen2.5-7B-Instruct	7.11	6.96	7.03	45.03	49.95	0.1095	0.6995	0.9442
w. DPO	7.37	6.96	7.17	50.55	55.00	0.1109	0.7061	0.9460
w. SimPO	7.41	6.98	7.20	48.76	52.75	0.1100	0.7047	0.9387
w. BPO	6.90	6.16	6.53	29.44	39.85	0.1086	0.6765	0.9418
w. BoN	7.50	7.10	7.30	50.42	55.43	0.1159	0.7066	0.9440
w. Aligner	6.24	3.76	5.00	42.15	45.82	0.1088	0.6993	0.9438
w. MetaAligner	6.41	5.13	5.77	36.58	38.45	0.0995	0.6966	0.9422
w. **DIFFPO**-2B	7.01	6.34	6.67	43.89	49.07	0.1074	0.7013	0.9659
w. **DIFFPO**-9B	7.62	7.10	7.35	57.89	63.01	0.1117	0.7100	0.9445
Qwen2.5-14B-Instruct	7.24	6.71	6.98	51.60	57.10	0.1117	0.7100	0.9445
w. BPO	7.21	6.82	7.02	37.02	47.51	0.1005	0.6853	0.9323
w. BoN	7.49	6.98	7.24	54.92	59.02	0.1182	0.7163	0.9485
w. Aligner	6.14	4.11	5.13	45.14	47.24	0.1114	0.7099	0.9438
w. MetaAligner	6.24	5.73	5.99	41.25	43.23	0.1092	0.7074	0.9375
w. **DIFFPO**-2B	7.08	6.33	6.71	43.70	48.76	0.1078	0.7017	0.9704
w. **DIFFPO**-9B	7.62	7.35	7.48	55.13	60.65	0.1136	0.7185	0.9759

is selected as $\mathbf{y}^{(T)}$. The remaining five responses are ranked according to their scores to serve as $\mathbf{y}^{(0:T-1)}$. In the training process, at each iteration, we randomly sample $\mathbf{y}^t$ from $\mathbf{y}^{(0:T-1)}$ for optimization. We train **DIFFPO** models using the following hyperparameters: a learning rate of 1e-9, a batch size of 1 and gradient accumulation steps of 4, a max sequence length of 1024, and a cosine learning rate schedule with 3% warmup steps for 1 epoch. All the models are trained with an Adam optimizer. All the training experiments in this paper were conducted on 8×A100 GPUs.

Evaluation Details. For the MT-bench, we use GPT-4 as the judge model, following the default settings. The scores are based on a single-answer rating scale from 1 to 10. For AlpacaEval, we use GPT-4 Turbo as the judge model, which performs pairwise comparison of responses generated by GPT-4, each with the same maximum length. For HH-RLHF, we use ArmoRM for single-answer rating and report the overall score, along with the "helpful" and "harmless" scores, which are provided in dimensions 9 and 10, respectively.

Baseline Details. Implementation details for different baselines are as follows:

- MetaAligner: we use the open-sourced MetaAligner-7B model https://huggingface.co/MetaAligner/MetaAligner-HH-RLHF-7B on Huggingface and follow its guideline on Huggingface.
- DPO, SimPO: we directly use open-sourced models https://huggingface.co/princeton-nlp on Huggingface.
- Args: We reproduce Args according to https://github.com/deeplearning-wisc/args/tree/main by replacing the reward model with ArmoRM (Wang et al., 2024).
- Aligner: we use the open-sourced Aligner-7B https://huggingface.co/aligner/aligner-7b-v1.0 on Huggingface and follow its guideline on Huggingface.
- BPO: we use the open-sourced BPO model https://huggingface.co/THUDM/BPO on Huggingface and follow its guideline on Huggingface.

B.2 Experimental Results

DIFFPO significantly outperforms existing preference optimization methods. We provided additional comparison with baselines, with results presented in Table 6. The experiments are conducted on base models of Qwen-2.5-7B and 14B. While all preference optimization algorithms improve performance over the base model, **DIFFPO** achieves the best overall performance

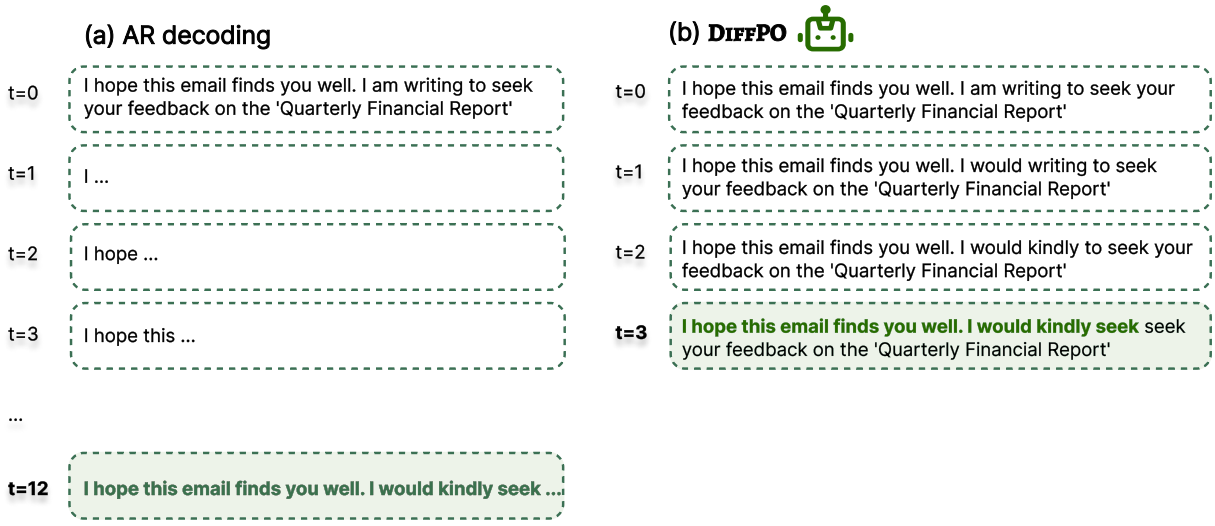

Figure 4: **Illustration of the Speedup of DiffPO.**

Table 7: **Loss and Hyperparameter Ablation.** We report the results of vanilla decoding from the base model and the optimized **DiffPO** model. The results indicate that applying **DiffPO** to the base model yields outperforming performance than single models, demonstrating the effectiveness of **DiffPO** strategy.

Loss	MT	AE2	HH
	GPT-4	LC (%)	Avg.
Vanilla Decoding			
Llama-3-Instruct	6.78	36.83	0.7985
L_{AR}	6.90	36.35	0.7971
$1,000 \times L_{\text{Con}} + L_{\text{AR}}$			
Llama-3-it w. **DiffPO**-8B			
L_{AR} only	6.75	35.84	0.7968
$10 \times L_{\text{Con}} + L_{\text{AR}}$	6.85	35.96	0.7997
$100 \times L_{\text{Con}} + L_{\text{AR}}$	6.86	35.92	0.7998
$1,000 \times L_{\text{Con}} + L_{\text{AR}}$	7.02	36.44	0.7998

across all benchmarks and settings. These consistent and significant improvements underscore the robustness and effectiveness of **DiffPO**. Notably, **DiffPO** outperforms the training-based baselines (i.e., SimPO and DPO) across various settings, despite requiring only a single training session of **DiffPO** model and being capable of enhancing the performance of multiple base models.

C Analysis

C.1 Loss and Hyperparameter Ablation.

We supplement the ablation results for **DiffPO**-8B, presented in Table 7. We report on two decoding strategies: *vanilla decoding* of the single base model and the optimized **DiffPO**-8B, and **DiffPO** *decoding*, which applies the optimized **DiffPO**-8B on the output of the base model. The findings indicate that applying **DiffPO** to the base model achieves performance superior to that of single models alone, thus demonstrating the effectiveness of the **DiffPO** strategy. Furthermore, we report the results of an ablation study on the hyperparameter w in Eq. 6. When using **DiffPO** decoding, employing L_{Con} with larger values of w lead to a more pronounced improvement in performance.

C.2 Illustration of the Speed-up of DiffPO

As shown in Figure 4, AR decoding (e.g., Aligner (Ji et al., 2024)) typically generates only one aligned token per iteration. In contrast, **DiffPO** enables the skipping of satisfied tokens, thereby avoiding the time latency associated with token-level generation. As a result, **DiffPO**can predict the modified subsequence in 3 iterations, achieving the same result as 11 iterations of AR decoding.