

EVOLVING LLMs’ SELF-REFINEMENT CAPABILITY VIA SYNERGISTIC TRAINING-INFERENCE OPTIMIZATION

006 **Anonymous authors**

007 Paper under double-blind review

ABSTRACT

013 *Self-Refinement* refers to a model’s ability to revise its own responses to produce
 014 improved outputs. This capability can also serve as a fundamental mechanism for
 015 *Self-Improvement*, for example by reconstructing datasets with refined results to
 016 enhance intrinsic model performance. However, our comprehensive experiments
 017 reveal that large language models (LLMs) show no clear evidence of inherent
 018 *Self-Refinement*; on average, response quality degrades over successive iterations.
 019 To address this gap, we propose **EVOLVE**, a simple yet effective framework for
 020 eliciting and tracking the evolution of *Self-Refinement* through iterative training.
 021 Moreover, we demonstrate the potential of leveraging *Self-Refinement* to achieve
 022 broader *Self-Improvement* of intrinsic model abilities. Experiments show that the
 023 evolved *Self-Refinement* ability enables the Llama-3.1-8B base model to surpass
 024 GPT-4o, achieving 62.3% length-controlled and 63.3% raw win rates on AlpacaEval
 025 2, and 50.3% on Arena-Hard. It also generalizes effectively to out-of-domain
 026 reasoning tasks, improving performance on mathematical reasoning benchmarks
 027 such as GSM8K and MATH.

1 INTRODUCTION

032 Large language models (LLMs) have demonstrated strong performance across a wide range of
 033 tasks through training on massive datasets (Achiam et al., 2023; Dubey et al., 2024). However, the
 034 supply of high-quality training data is becoming increasingly scarce, limiting further progress. As a
 035 complementary direction, *Self-Improvement* methods seek to enhance a model’s intrinsic capabilities
 036 by leveraging model-generated data and feedback with minimal external supervision (Tao et al.,
 037 2024; Huang et al., 2022). A critical challenge in this field is ensuring the quality and robustness
 038 of synthetic datasets while minimizing noise, often by leveraging intrinsic model mechanisms or
 039 properties. To this end, we investigate inference-time *Self-Refinement*, where a model revises its own
 040 outputs to improve response accuracy and stability. Previous work has investigated incorporating
 041 *Self-Refinement* strategies into broader frameworks, ranging from prompt-based techniques that let
 042 models iteratively revise their own drafts (Madaan et al., 2024; Paul et al., 2023) to approaches that
 043 incorporate external feedback, such as preference signals or additional contextual cues (Stiennon
 044 et al., 2020; Asai et al., 2024). These efforts raise a fundamental question: ***Can LLMs autonomously***
refine their responses without external information, thereby achieving Self-Refinement?

045 To investigate this question, we conducted preliminary experiments to assess whether contemporary
 046 LLMs possess an inherent *Self-Refinement* capability. To mitigate potential biases from prompt design,
 047 we evaluated three distinct refinement templates (detailed in Appendix J), spanning a spectrum from
 048 detailed guidance to minimal intervention: (1) a guided template requiring direct output of an
 049 improved response, emphasizing clarity, accuracy, and conciseness without analysis; (2) a guided
 050 template mandating an initial analysis of the example response’s strengths and weaknesses, followed
 051 by the refined output; and (3) a minimalist template with no explicit instructions, to minimize prompt-
 052 induced artifacts. These templates were applied across diverse LLMs with varying architectures,
 053 enabling iterative *Self-Refinement* cycles. As shown in Fig. 1, our results reveal no clear evidence of
 inherent *Self-Refinement*; on average, response quality degrades over successive iterations.

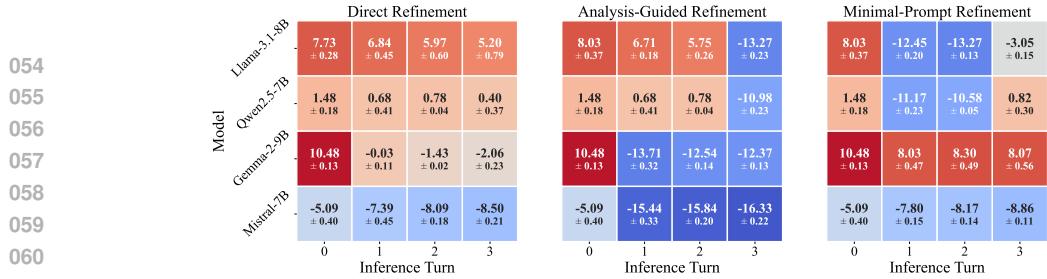


Figure 1: Evaluation of *Self-Refinement* Capability Across Various Models. We use three refinement templates to minimize prompt bias. The x-axis denotes the inference iteration number. For each turn, responses are generated from 256 UltraFeedback test set samples, using the original prompt and the prior turn’s output. These are then scored by the Skywork Reward Model (Liu et al., 2024). To eliminate potential randomness, the reported values are the mean score of three independent runs with different random seeds; higher scores indicate better quality. Templates are detailed in Appendix J.

Motivated by these findings, we move beyond diagnosis to ask: *can this capability be effectively activated and strengthened through training, and how might it be leveraged for sustained Self-Improvement?* Specifically, we first investigate fine-tuning methods to activate the *Self-Refinement* capabilities of LLMs. Building on this, we further explore strategies to continuously enhance LLMs’ *Self-Refinement* abilities, encompassing ongoing data acquisition and iterative training updates. To analyze the evolution of *Self-Refinement* capabilities in stages, we conduct our study within an iterative preference training framework. Finally, we examine potential applications of activated *Self-Refinement* capability, such as leveraging it to achieve *Self-Improvement* of the model’s intrinsic capabilities. The contributions of our work are summarized as follows:

- We conduct a comprehensive study on eliciting and enhancing *Self-Refinement* capability through iterative training, including proposing a novel training method to generate improved responses from previous outputs, and exploring diverse strategies for effectively leveraging *Self-Refinement* to collect datasets.
- We introduce EVOLVE, a simple yet effective framework for analyzing the evolution of *Self-Refinement* ability throughout iterative training cycles. Furthermore, we investigate the potential of leveraging *Self-Refinement* to achieve *Self-Improvement* of model intrinsic abilities.
- We empirically validate the effectiveness of our framework in enhancing *Self-Refinement* capability. Starting from the Llama-3.1-8B base model, EVOLVE outperforms Llama-3.1-405B-Instruct and GPT-4o, achieving 62.3% length-controlled and 63.3% raw win rates on AlpacaEval 2, along with 50.3% on Arena-Hard, while also improving performance on mathematical reasoning benchmarks such as GSM8K and MATH.

2 RELATED WORKS

LLM Self-Improvement. *Self-Improvement* in LLMs aims to enhance intrinsic model capabilities with minimal external supervision, addressing the challenge of scarce high-quality training data (Tao et al., 2024; Huang et al., 2022). Recent approaches leverage model-generated data to iteratively improve model performance, often through synthetic dataset construction or feedback-driven optimization (Wang et al., 2022a;b; 2025). For instance, self-consistency decoding (Wang et al., 2022a) enables models to sample multiple outputs and select the most consistent, improving performance in reasoning tasks, while methods like Self-Instruct (Wang et al., 2022b) automate instruction-tuning data creation to enhance task generalization. Other works explore iterative bootstrapping, where models refine their outputs over multiple rounds (Madaan et al., 2024; Huang et al., 2022), or employ internal scoring mechanisms to filter and improve dataset quality (Yuan et al., 2024). Our work builds on these foundations, focusing on activating and enhancing *Self-Refinement* as a mechanism for sustained *Self-Improvement*, distinct from prior approaches by emphasizing iterative training to strengthen intrinsic refinement capabilities.

Data Generation and Iterative Optimization. The scarcity of high-quality training data has motivated the development of new data-generation strategies to enable sustainable iterative optimization (Long et al., 2024; Ding et al., 2024). Techniques such as data augmentation through LLM-generated synthetic datasets have proven effective in scaling training data for smaller models (Chen et al., 2023; Xu et al., 2024; Taori et al., 2023). Recent studies further explore online data generation, where

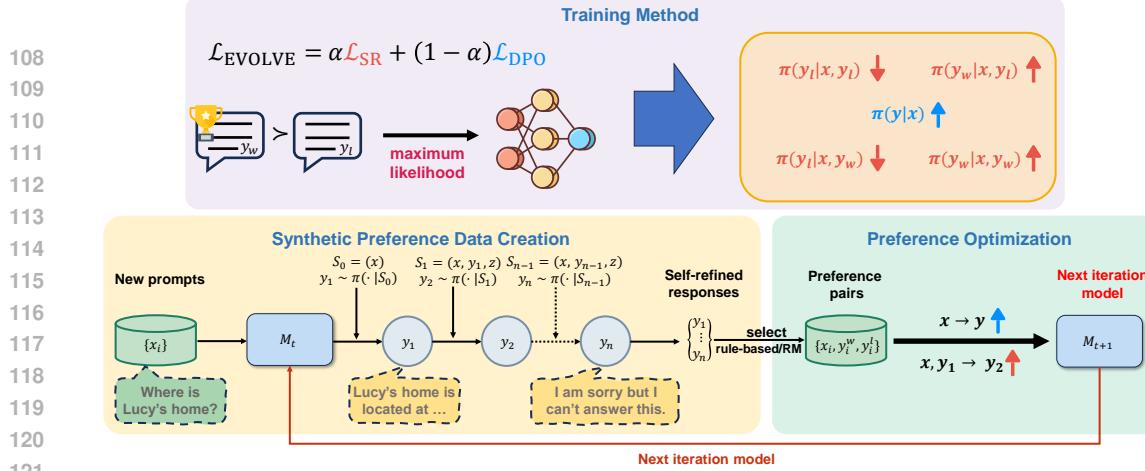


Figure 2: Our framework, **EVOLVE**, iteratively alternates between inference and training processes. In iteration t , Model M_t uses the *Self-Refinement* strategy to collect preference data, which is then utilized to enhance the model’s intrinsic capabilities via preference training (Eq. 5), yielding the next iteration model M_{t+1} . The dataset is filtered through either a rule-based method or a reward model.

models dynamically generate and refine datasets during training (Tian et al., 2024; Xiong et al., 2023). In this work, we develop a closed-loop system that harnesses models’ *Self-Refinement* capabilities to autonomously generate graded data, enabling iterative model optimization.

3 METHODOLOGY

In this section, we first investigate the training method for activating *Self-Refinement* capabilities in LLMs. Building on the activated *Self-Refinement* abilities, we further investigate approaches to progressively enhance the *Self-Refinement* capacities of LLMs, encompassing both data acquisition and iterative training components. Drawing on these findings, we propose a simple yet effective framework **EVOLVE** for iterative training and inference to study the evolution of *Self-Refinement* capabilities during the iterative training process.

Evaluation Protocol. To substantiate our analysis below, we now describe the evaluation protocol employed in this section. For each generation round, we use 256 samples from the UltraFeedback (Lambert et al., 2024a) test set to generate responses. These responses are then scored using the Skywork Reward Model (Liu et al., 2024), a compact yet high-performing model on the RewardBench leaderboard (Lambert et al., 2024b). Results are reported as average scores.

3.1 TRAINING STAGE OF EVOLVE: SYNERGISTIC OPTIMIZATION OF SFT AND PT

Starting from a base pretrained language model, we investigate post-training fine-tuning methods, covering a synergistic optimization process of Supervised Fine-Tuning (SFT) and Preference Training (PT) stages, to activate and enhance the *Self-Refinement* capability of LLMs. Specifically, SFT serves to initiate *Self-Refinement*, enabling the model to learn how to revise suboptimal responses, while PT further strengthens this ability, improving the model’s proficiency in applying it during iterative optimization. These two stages complement each other, forming the foundation of our framework. To support this claim, we first present the experimental results of the different training configurations tested for the SFT and PT stages, as shown in Fig. 3.

Supervised Fine-Tuning (SFT) for Activating Self-Refinement. Our experimental findings reveal that the SFT stage is fundamental for instilling the *Self-Refinement* capability, enabling the effective operation of the entire framework. Building on the standard Negative Log-Likelihood (NLL) SFT loss, we introduce specific modifications to explicitly encourage the model to refine suboptimal

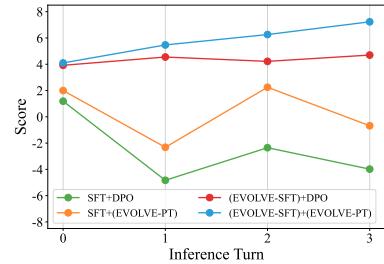


Figure 3: Ablation of training combinations. SFT activates *Self-Refinement*, PT enhances it, and their synergy (blue, ours) yields the best performance.

162 responses:

163
$$\mathcal{L}_{\text{EVOLVE-SFT}}(\pi_\theta) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} [\log \pi_\theta(y_w|x) + \log \pi_\theta(y_w|x, y_l, z)], \quad (1)$$
 164

165 where (x, y_w, y_l) is sampled from a preference dataset \mathcal{D} , x represents the prompt, y_w is the preferred 166 response, and y_l is the rejected response. The variable z denotes the refinement template, as shown in 167 Appendix D.1.168 Without these SFT adaptations, the model fails to adequately learn *Self-Refinement* behavior. This is 169 reflected in Fig. 3, where the response quality oscillates significantly across inference turns. 170171 **Preference Training (PT) for Strengthening Self-Refinement.** PT stage is designed to further 172 enhance the model’s *Self-Refinement* capability based on the SFT phase, enabling it to more effectively 173 master and apply this skill. To improve the model’s ability to refine an initial response into a better 174 one during PT, we first consider the following scenario: given a prompt x and an initial response y_1 , 175 how can the model be guided to produce a refined response y_2 ? We model this problem as follows:

176
$$\max_{\pi} \mathbb{E}_{y_2 \sim \pi(\cdot|x, y_1, z)} \left[p(y_2 \succ y_1|x) - \beta D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}|x, y_1, z) \right]. \quad (2)$$
 177

178 where $p(y_2 \succ y_1|x)$ is the preference function, denoting the probability that y_2 is preferred over y_1 179 given x . By deriving the optimal policy from Eq. 2 and applying a mean squared error formulation, 180 we parameterize the policy model as π_θ , yielding:

181
$$\mathcal{L}(\pi_\theta; \pi_{\text{ref}}) = \mathbb{E}_{(x, y_1, y_2) \sim \rho} \left[\log \left(\frac{\pi_\theta(y_2|x, y_1, z) \pi_{\text{ref}}(y_1|x, y_1, z)}{\pi_\theta(y_1|x, y_1, z) \pi_{\text{ref}}(y_2|x, y_1, z)} \right) - \left(\frac{p(y_2 \succ y_1|x)}{\beta} - \frac{1}{2\beta} \right) \right]^2, \quad (3)$$
 182

183 where ρ denotes the true distribution (derivation in Appendix C.1). Given a curated preference dataset 184 $\mathcal{D} = \{(x^{(i)}, y_w^{(i)}, y_l^{(i)})\}_{i=1}^N$ with $y_w \succ y_l$, we further define the final *Self-Refinement* loss:

185
$$\mathcal{L}_{\text{SR}}(\pi_\theta; \pi_{\text{ref}}) = \mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\frac{1}{2} - v(x, y_l, y_w, z; \pi_\theta) \right]^2 + \mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\frac{1}{2} + v(x, y_w, y_l, z; \pi_\theta) \right]^2, \quad (4)$$
 186

187 where $v(x, y_1, y_2, z; \pi_\theta) = \beta \log \left(\frac{\pi_\theta(y_2|x, y_1, z) \pi_{\text{ref}}(y_1|x, y_1, z)}{\pi_\theta(y_1|x, y_1, z) \pi_{\text{ref}}(y_2|x, y_1, z)} \right)$. Finally, we combine the *Self- 188 Refinement* loss with the DPO loss (Rafailov et al., 2023), obtaining:

189
$$\mathcal{L}_{\text{EVOLVE-PT}}(\pi_\theta; \pi_{\text{ref}}) = \alpha \mathcal{L}_{\text{SR}}(\pi_\theta; \pi_{\text{ref}}) + (1 - \alpha) \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}). \quad (5)$$
 190

191 The effectiveness of the PT stage is inherently tied to the foundational adjustments made during 192 SFT, while the SFT phase requires the continuous activation of the *Self-Refinement* capability from 193 the PT stage, demonstrating that **the two phases are truly complementary**. As shown in Fig. 3, 194 without the refinements introduced in the PT stage, the *Self-Refinement* capability acquired during 195 SFT remains limited. Thus, only through the synergistic optimization of both stages can the model’s 196 *Self-Refinement* ability be effectively activated and consistently strengthened. Furthermore, as 197 discussed in Appendix E, we explore alternative *Self-Refinement* objective formulations in the PT 198 stage, observing comparable improvements. This suggests that the development of *Self-Refinement* 199 stems from inherent advantages of the overall training paradigm, rather than a particular loss design.200

3.2 INFERENCE STAGE OF EVOLVE: EXPLORING DYNAMIC GENERATION STRATEGIES

 201202 Building on the training stage, we next examine how to exploit the acquired *Self-Refinement* capability 203 during inference. To identify the most suitable strategy for our final framework, we systematically 204 summarize and compare four distinct generation strategies:

- **Parallel Sampling:** Given a problem x , the model generates multiple candidate answers $\{y_1, y_2, \dots, y_n\}$ independently in parallel.
- **Chain of Self-Refinement:** For a problem x , the model first generates an initial answer y_1 . Using the refinement template z , it then iteratively produces refined responses y_n based on x and the previous response y_{n-1} .
- **Few-Shot Self-Refinement:** The model iteratively improves its outputs y_n by conditioning on both the input x and all prior generations $\{y_1, y_2, \dots, y_{n-1}\}$.
- **Self-Refinement with Self-Evaluation:** The model first generates an initial response y_1 for the problem x , then evaluates it along multiple dimensions (e.g., relevance, helpfulness) to produce an evaluation e_1 . At each subsequent step, a refined response y_n is generated based on x , the previous response y_{n-1} , and the last evaluation e_{n-1} : $y_n \sim \pi_\theta(\cdot|x, y_{n-1}, e_{n-1})$. Each evaluation e_n is generated from (x, y_n) .

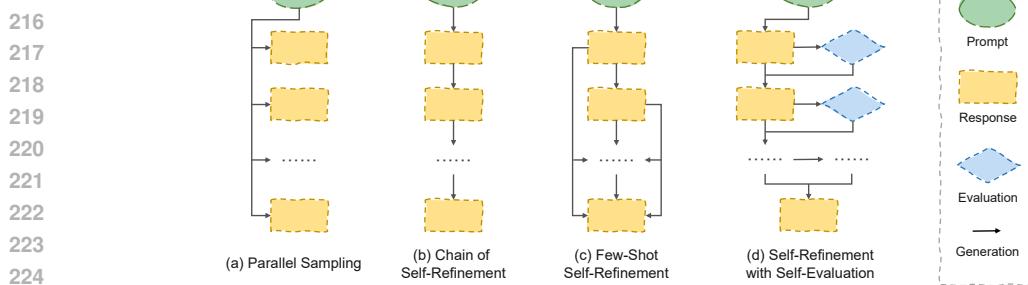


Figure 4: Illustration of four dynamic generation strategies.

The procedural logic of four generation strategies is presented in Fig. 4 and corresponding prompt templates are provided in Appendix K. Based on the experimental results shown in Fig. 5, we can make the following observations: *Parallel Sampling* maintains stable mean rewards across iterations, as all outputs are drawn from the same distribution. *Chain of Self-Refinement* achieves the best performance, consistently improving response quality across iterations. *Few-Shot Self-Refinement* exhibits an initial increase but eventually degrades, as longer prompts hinder the model’s ability to capture key information. *Self-Refinement with Self-Evaluation* also yields iterative gains but remains weaker than *Chain of Self-Refinement*, primarily due to the model’s limited self-evaluation capability, as illustrated in Appendix K. Since **Chain of Self-Refinement best aligns with our training approach, consistently improves response quality, and incurs only minimal overhead**, we adopt it as the core generation strategy in our framework.

In summary, by integrating insights from both the training and inference phases, we propose EVOLVE, a streamlined and effective framework, as illustrated in Fig. 2. Our EVOLVE framework iteratively conducts preference training and data generation. This iterative training process continually enhances the LLMs’ *Self-Refinement* capability, which in turn enables the hierarchical generation of high-quality data. Through this mutually reinforcing mechanism, EVOLVE achieves sustained improvement in a unified training loop.

4 EXPERIMENTS

Building on the evaluation in Fig. 1, where current LLMs show no clear evidence of inherent *Self-Refinement*, we propose the EVOLVE framework to enhance this capability. In this section, we demonstrate its effectiveness by addressing three key questions:

1. How much does EVOLVE improve model performance compared to prior methods (§4.2)?
2. Which components are responsible for the performance improvements of EVOLVE (§4.3)?
3. Does the *Self-Refinement* ability induced by EVOLVE generalize to out-of-domain tasks (§4.4)?

4.1 EXPERIMENTS SETTING

Models and Training Settings. Our experiments were conducted on the Llama-3.1-8B Base model (Dubey et al., 2024) and Mistral-7B Base model (Jiang et al., 2023). During the **SFT phase**, we fine-tune the base model on the Llama-3.1-tulu-3-70b-preference-mixture dataset (Lambert et al., 2024a) using Eq. 1, yielding the EVOLVE-SFT model. In the **preference training phase**, we continue to train the EVOLVE-SFT model using 30K preference pairs from the UltraFeedback dataset (Cui et al., 2023) with Eq. 5, obtaining the EVOLVE *offline* model. For **online iterative training**, we construct new preference datasets through *Self-Refinement*: (1) we first sample 5K prompts from UltraFeedback, generate four responses per prompt with the EVOLVE *offline* model using iterative *Self-Refinement*, forming a dataset $\mathcal{D} = \{x^{(i)}, y_1^{(i)}, y_2^{(i)}, y_3^{(i)}, y_4^{(i)}\}$; (2) we then score these responses with the Skywork Reward Model (Liu et al., 2024), which is a compact yet high-performing model on the RewardBench leaderboard (Lambert et al., 2024b); (3) we construct a new preference dataset and train a new model, denoted EVOLVE *iter1*. Finally, we repeat the same procedure with another 10K prompts from UltraFeedback, producing the EVOLVE *iter2* model.

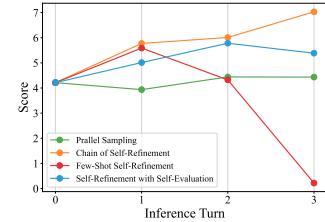


Figure 5: Performance of four generation strategies. *Chain of Self-Refinement* achieves the best results across iterations.

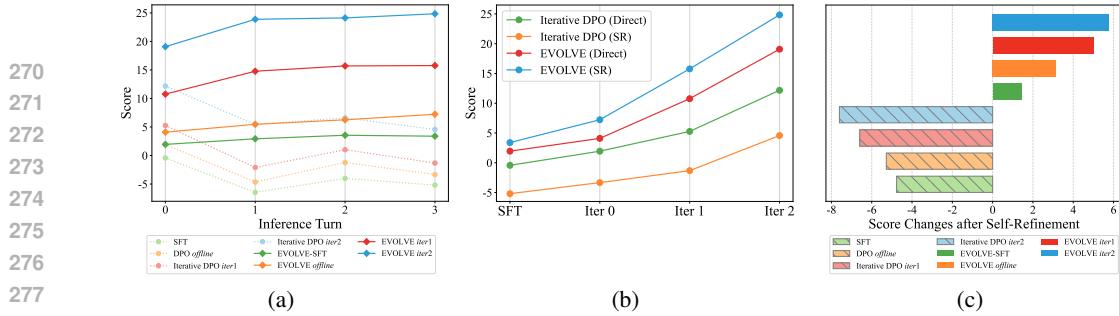


Figure 6: Evolution of *Self-Refinement* Capability through Iterative Training. (a) shows the evolution of *Self-Refinement* capability for each iteratively trained model during the inference stage. (b) depicts the performance progression of both the **Direct** and **SR** methods throughout iterative training. (c) demonstrates the performance improvement brought by **SR** compared to **Direct** across training iterations. Responses are generated from 256 UltraFeedback test set samples.

Baselines. We first train the base model by applying standard SFT and DPO losses, following the same procedure and dataset as in our method, to obtain the SFT model and the DPO *offline* model. Based on these, we compare against several state-of-the-art baselines: (1) **Iterative DPO** (Snorkel, 2024; Xiong et al., 2023), an iterative preference optimization method; (2) **SynPO** (Dong et al., 2024), which leverages a refiner model to enhance performance; (3) **SRPO** (Choi et al., 2024), an offline preference optimization approach guided by an adversarial objective to improve response quality iteratively; and (4) **ScoRe** (Kumar et al., 2024), an online reinforcement learning approach for self-correction. Iterative DPO and SynPO are trained using DPO method, while SRPO is adapted for our iterative setting. Since ScoRe is designed for reasoning tasks, we integrate it into general tasks and perform RL training on the EVOLVE-SFT model. The specific implementation details of these methods are provided in Appendix B.

Evaluation Benchmarks. We evaluate our models on AlpacaEval 2 (Li et al., 2023) and Arena-Hard (Li et al., 2024), as well as on two cross-domain mathematical reasoning tasks: GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021). AlpacaEval 2 consists of 805 questions drawn from five datasets, while Arena-Hard comprises 500 well-defined technical questions. GSM8K contains elementary- and middle-school-level math problems, whereas MATH includes more challenging questions spanning diverse mathematical branches such as algebra, counting and probability, geometry, number theory, and calculus. For AlpacaEval 2, we report both the raw win rate (WR) and the length-controlled (LC) win rate (Dubois et al., 2024). For Arena-Hard, we measure the win rate relative to the baseline model. For GSM8K and MATH, we adopt the Step-DPO evaluation script (Lai et al., 2024). We assess all models under two settings: direct response generation (**Direct**) and responses refined through three rounds of *Self-Refinement* (**SR**).

4.2 BOOST IN MODEL PERFORMANCE WITH EVOLVE

Consistent Gains in Inference and Iterative Training. We begin by investigating the evolution of *Self-Refinement* capability during iterative training on the UltraFeedback test set, as shown in Fig. 6. EVOLVE demonstrates that *Self-Refinement* is not only activated but also progressively strengthened across iterations. As shown in Fig. 6(a) and Fig. 6(b), models trained with our framework continually improve during inference, while their refinement ability evolves steadily throughout iterative training. The training algorithm and *Self-Refinement* strategy complement each other to form a synergistic training loop that drives continual gains. Fig. 6(c) further shows that the gain from *Self-Refinement* over direct response generation increases with each iteration, confirming that EVOLVE enables models to gradually internalize and reinforce this cognitive mode.

Superior Benchmark Performance over Baselines. On AlpacaEval 2 and Arena-Hard (Tab. 1), EVOLVE achieves consistent improvements: it significantly boosts performance in the **Direct** setting and delivers a qualitative leap under the **SR** setting, demonstrating the benefits of progressive refinement. In contrast, preference-optimization methods built upon SFT often fail to enable effective *Self-Refinement*, even when the training objective explicitly includes it. Although ScoRe also shows strong refinement ability, RL-based approaches that rely on reward model scores frequently suffer from instability in general domains, largely due to reward hacking that undermines reliable optimization. These results confirm that EVOLVE provides more stable and comprehensive gains across benchmarks compared to existing methods.

Table 1: Performance comparison across AlpacaEval 2 and Arena-Hard benchmarks under the **Direct** and **SR** settings. Reported values are scores, with parentheses indicating gains over SFT baseline.

Method	Direct			Self-Refinement (SR)		
	AlpacaEval 2		Arena-Hard	AlpacaEval 2		Arena-Hard
	LC (%)	WR (%)	WR (%)	LC (%)	WR (%)	WR (%)
Llama-3.1-8B Base						
SFT	15.9	12.7	12.7	13.8	8.1	8.0
+SRPO <i>offline</i>	15.8 (-0.1)	16.0 (+3.3)	14.3 (+1.6)	22.2 (+8.4)	16.2 (+8.1)	13.2 (+5.2)
+SRPO <i>iter1</i>	21.9 (+6.0)	21.8 (+9.1)	21.9 (+9.2)	27.1 (+13.3)	20.2 (+12.1)	18.8 (+10.8)
+SRPO <i>iter2</i>	22.1 (+6.2)	22.4 (+9.7)	25.6 (+12.9)	27.5 (+13.7)	22.3 (+14.2)	21.8 (+13.8)
+DPO <i>offline</i>	17.9 (+2.0)	16.7 (+4.0)	16.5 (+3.8)	18.3 (+4.5)	12.6 (+4.5)	12.6 (+4.6)
+SynPO <i>iter1</i>	23.5 (+7.6)	23.1 (+10.4)	21.6 (+8.9)	22.5 (+8.7)	15.1 (+7.0)	14.0 (+6.0)
+SynPO <i>iter2</i>	23.1 (+7.2)	18.2 (+5.5)	19.6 (+6.9)	20.6 (+6.8)	13.8 (+5.7)	16.8 (+8.8)
+Iterative DPO <i>iter1</i>	24.6 (+8.7)	22.3 (+9.6)	22.4 (+9.7)	25.1 (+11.3)	16.9 (+8.8)	17.1 (+9.1)
+Iterative DPO <i>iter2</i>	34.1 (+18.2)	33.5 (+20.8)	29.6 (+16.9)	34.5 (+20.7)	27.8 (+19.7)	23.7 (+15.7)
EVOLVE-SFT (Ours)	15.9 (+0.0)	15.5 (+2.8)	16.5 (+3.8)	20.0 (+6.2)	18.6 (+10.5)	18.0 (+10.0)
+ScoRe	21.3 (+5.4)	27.9 (+15.2)	24.0 (+11.3)	35.8 (+22.0)	42.7 (+34.6)	34.0 (+26.0)
+EVOLVE <i>offline</i> (Ours)	19.1 (+3.2)	18.6 (+5.9)	17.4 (+4.7)	28.8 (+15.0)	27.1 (+19.0)	23.5 (+15.5)
+EVOLVE <i>iter1</i> (Ours)	32.7 (+16.8)	33.5 (+20.8)	31.9 (+19.2)	50.2 (+36.4)	49.9 (+41.8)	37.5 (+29.5)
+EVOLVE <i>iter2</i> (Ours)	45.0 (+29.1)	46.8 (+34.1)	38.0 (+25.3)	62.3 (+48.5)	63.3 (+55.2)	50.3 (+42.3)
Mistral-7B Base						
SFT	17.0	15.8	11.7	9.2	5.6	4.0
+DPO <i>offline</i>	19.0 (+2.0)	19.3 (+3.5)	13.8 (+2.1)	13.0 (+3.8)	10.6 (+5.0)	3.9 (-0.1)
+Iterative DPO <i>iter1</i>	29.7 (+12.7)	33.5 (+17.7)	22.8 (+11.1)	15.3 (+6.1)	17.6 (+12.0)	13.7 (+9.7)
+Iterative DPO <i>iter2</i>	39.5 (+22.5)	41.8 (+26.0)	27.2 (+15.5)	7.3 (-1.9)	11.2 (+5.6)	21.1 (+17.1)
EVOLVE-SFT (Ours)	18.8 (+1.8)	17.4 (+1.6)	11.1 (-0.6)	20.5 (+11.3)	18.2 (+12.6)	12.9 (+8.9)
+EVOLVE <i>offline</i> (Ours)	16.2 (-0.8)	16.7 (+0.9)	13.3 (+1.6)	23.3 (+14.1)	22.7 (+17.1)	15.6 (+11.6)
+EVOLVE <i>iter1</i> (Ours)	29.1 (+12.1)	30.2 (+14.4)	21.9 (+10.2)	39.0 (+29.8)	40.6 (+35.0)	25.3 (+21.3)
+EVOLVE <i>iter2</i> (Ours)	39.3 (+22.3)	40.2 (+24.4)	27.7 (+16.0)	46.4 (+37.2)	49.7 (+44.1)	32.8 (+28.8)

4.3 IMPACT OF KEY COMPONENTS ON EVOLVE PERFORMANCE

We analyze three critical factors in EVOLVE: (1) the role of the DPO loss in the training objective, (2) the importance of the *Self-Refinement* generation strategy for data collection, (3) the framework’s dependence on the choice of reward model. The corresponding results are presented in Fig. 7, Fig. 8 and Appendix I.2.

Impact of DPO Loss. As shown in Fig. 7, omitting the DPO loss progressively degrades the performance of EVOLVE. With iterative training, the absence of DPO loss results in a slower improvement rate in both the **Direct** and **SR** settings. This highlights the crucial role of DPO loss in strengthening the model’s ability to generate higher-quality responses directly.

Impact of Self-Refinement Generation Strategy. Fig. 8 shows that models trained on datasets collected via *Parallel Sampling* (PS) consistently underperform compared to those using the *Self-Refinement* generation strategy. This demonstrates that the *Self-Refinement* generation strategy is essential for constructing high-quality preference datasets. Further evidence is provided in Appendix L, where we show that *Self-Refinement* improves responses by enhancing logical coherence and producing clearer, more concise expressions.

Impact of Reward Model Selection. To assess the influence of reward models, we conducted ablation studies using the Skywork Reward Model (Liu et al., 2024) and ArmoRM (Wang et al., 2024). The results, shown in Tab. 10, demonstrate that models trained with ArmoRM exhibit performance comparable to those trained with the Skywork Reward Model. This indicates that our framework’s effectiveness is independent of the reward model, and its ability to consistently enhance *Self-Refinement* capabilities remains robust across different reward model choices.

4.4 GENERALIZATION OF EVOLVE TO OUT-OF-DOMAIN REASONING TASKS

The results in Tab. 2 show that the *Self-Refinement* ability activated by EVOLVE generalizes effectively to reasoning tasks, yielding measurable performance gains on GSM8K and MATH.

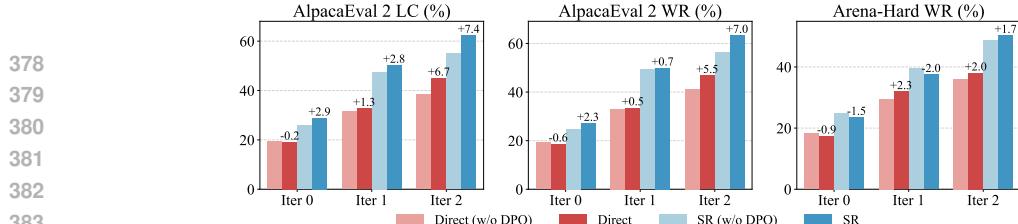


Figure 7: Impact analysis of DPO Loss on EVOLVE Performance.

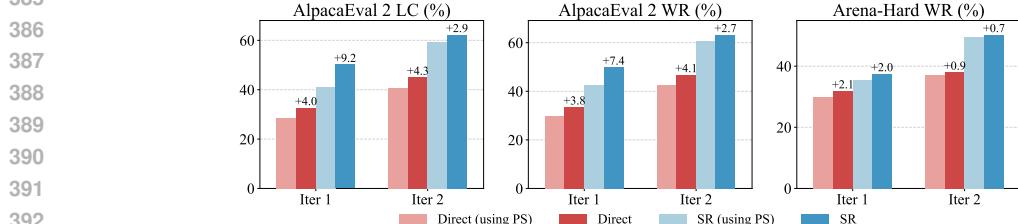
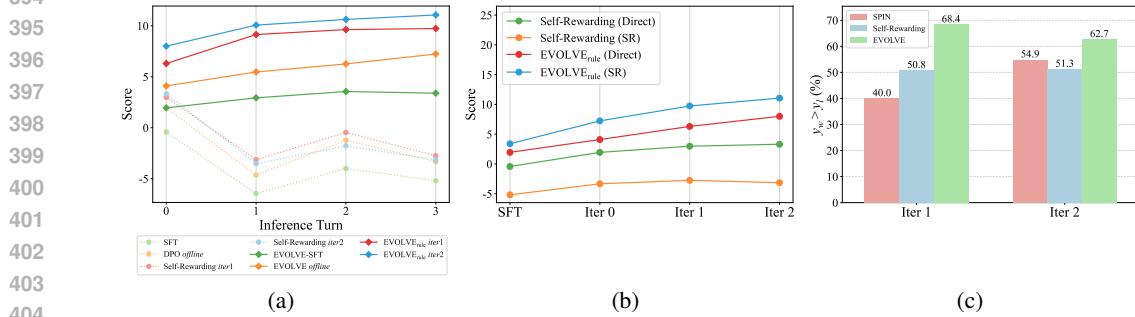
Figure 8: Effect of *Parallel Sampling* (PS) vs. *Self-Refinement* on EVOLVE Performance.

Figure 9: Model performance evaluation under the Self-Improvement setup: (a) shows performance variation with increasing inference turns. (b) depicts improvement during iterative training. (c) assesses the collected dataset quality using Skywork Reward Model.

This demonstrates that our method can transfer beyond its training domain and enhance reasoning ability without explicit exposure to mathematical data. For comparison, *ScoRe* also exhibits strong generalization on these tasks, which can be attributed to the inherent robust generalization properties of RL. However, as model accuracy on *GSM8K* and *MATH* increases, the improvements from *Self-Refinement* become less pronounced. We attribute this to the absence of domain-specific training, which prevents the model from mastering certain advanced problem-solving strategies. This observation motivates future work on incorporating domain-specific training for reasoning tasks, with the goal of further strengthening the reasoning capabilities of *EVOLVE*.

Table 2: Accuracy of Various Methods on *GSM8K* and *MATH* with *Llama-3.1-8B*.

Method	<i>GSM8K</i> (%)		<i>MATH</i> (%)	
	Direct	SR	Direct	SR
SFT	60.9	62.3	18.2	29.5
+SRPO <i>offline</i>	66.7	63.9	36.2	34.5
+SRPO <i>iter1</i>	67.2	66.6	38.5	37.1
+SRPO <i>iter2</i>	65.0	68.7	41.5	39.3
+DPO <i>offline</i>	66.3	66.6	35.1	34.6
+SynPO <i>iter1</i>	67.6	66.9	38.0	37.0
+SynPO <i>iter2</i>	67.6	66.3	33.2	32.4
+Iterative DPO <i>iter1</i>	68.5	67.9	36.6	35.6
+Iterative DPO <i>iter2</i>	68.4	67.2	36.7	36.3
<i>EVOLVE-SFT</i>	64.3	70.9	32.8	45.5
+ <i>ScoRe</i>	64.0	71.7	39.5	50.1
+ <i>EVOLVE</i> <i>offline</i>	68.3	72.2	38.3	42.4
+ <i>EVOLVE</i> <i>iter1</i>	70.2	71.7	46.1	47.8
+ <i>EVOLVE</i> <i>iter2</i>	71.9	73.6	48.7	50.1

5 EXTENSION: THE POTENTIAL OF SELF-REFINEMENT FOR ACHIEVING MODEL SELF-IMPROVEMENT

Building on the discussion of activating and enhancing *Self-Refinement* capabilities, we now explore a compelling question: *Can the activated Self-Refinement ability of LLMs enable model intrinsic performance Self-Improvement?* We investigate this using our *EVOLVE* framework.

Table 3: Performance Analysis of *Self-Refinement* Potential for *Self-Improvement*. Results evaluated on AlpacaEval 2 (Li et al., 2023) and Arena-Hard (Li et al., 2024) under the **Direct** and **SR** settings. **Direct** denotes direct response generation, **SR** indicates three rounds of *Self-Refinement* on responses.

Method	Direct			Self-Refinement (SR)		
	AlpacaEval 2		Arena-Hard	AlpacaEval 2		Arena-Hard
	LC (%)	WR (%)	WR (%)	LC (%)	WR (%)	WR (%)
Llama-3.1-8B Base						
SPIN <i>iter1</i>	15.5	12.8	11.1	17.2	12.4	10.2
SPIN <i>iter2</i>	13.2	11.3	12.1	12.3	9.9	11.9
Self-Rewarding <i>iter1</i>	19.3	17.2	14.2	19.3	12.6	11.6
Self-Rewarding <i>iter2</i>	18.2	14.2	15.9	19.0	11.2	11.0
EVOLVE _{rule} <i>iter1</i>	<u>23.9</u>	<u>24.5</u>	<u>22.0</u>	<u>37.4</u>	<u>35.7</u>	<u>31.1</u>
EVOLVE _{rule} <i>iter2</i>	28.4	29.7	24.9	41.3	39.5	32.0
Mistral-7B Base						
SPIN <i>iter1</i>	14.8	13.3	7.2	8.6	7.5	3.3
SPIN <i>iter2</i>	12.9	9.7	8.0	6.0	3.3	3.4
Self-Rewarding <i>iter1</i>	21.0	18.0	13.2	13.0	9.5	4.7
Self-Rewarding <i>iter2</i>	<u>21.9</u>	<u>20.8</u>	<u>12.4</u>	<u>8.5</u>	<u>7.9</u>	<u>4.0</u>
EVOLVE _{rule} <i>iter1</i>	19.7	<u>21.0</u>	<u>14.2</u>	28.2	<u>28.7</u>	<u>17.9</u>
EVOLVE _{rule} <i>iter2</i>	22.2	24.2	18.2	<u>27.5</u>	29.2	21.4

Rules for Achieving Model Self-Improvement. To objectively assess the model’s *Self-Refinement* ability, we employ a streamlined rule-based method for dataset filtering in this section. Specifically, for a given problem x , the model generates a sequence of responses $\{y_1, y_2, y_3, y_4\}$. We then directly designate y_1 as the rejected response, while y_4 as the chosen response. These pairs are used to construct the preference dataset for subsequent preference training.

Analysis Setup. In this section, we focus on **online iterative training**. We select Self-Rewarding (Yuan et al., 2024) and SPIN (Chen et al., 2024) as baselines, which build upon the DPO *offline* model and are further trained using the same 5K-prompt dataset to produce the *iter1* model, followed by a 10K-prompt dataset to yield the *iter2* model, as described in Section 4.1. In contrast, our approach employs the **rule-based method** described above for data filtering, relying on the model’s intrinsic *Self-Refinement* capability without external supervision. Starting from the EVOLVE *offline* model, we conduct two rounds of online training to obtain the EVOLVE *iter1* and EVOLVE *iter2* models.

Evolution of Self-Refinement and Dataset Augmentation We first investigate the evolution of *Self-Refinement* capability during iterative training for model *Self-Improvement* and analyze the quality of self-collected datasets, as shown in Fig. 9. Similar to experiments using reward models for data filtering, we observe that EVOLVE-trained models achieve synergistic improvements in both direct answering and *Self-Refinement* capabilities during iterative training. However, under unsupervised conditions, the rate of capability improvement slows. Additionally, we compare the quality of datasets collected by different methods, revealing that *Self-Refinement* enables superior performance gains by facilitating the collection of higher-quality datasets, as shown in Fig. 9(c).

Challenges with Self-Rewarding and SPIN. As shown in Table 3, Self-Rewarding and SPIN exhibit performance fluctuations and occasional declines, primarily due to the quality of their collected datasets. Self-Rewarding suffers from low dataset discriminability, resulting in stagnant performance. SPIN experiences further degradation due to even lower dataset quality, highlighting the challenges of achieving *Self-Improvement* in data-scarce domains.

Summary. *Self-Refinement* significantly enhances model performance by collecting higher-quality datasets without external supervision. However, experiments show it also introduces significant noise, highlighting the field’s complexity and the need for further study. In Appendix F, we further explore the upper bounds of *Self-Improvement*, offering deeper insights into its potential.

6 CONCLUSION

We present EVOLVE, a novel framework that integrates iterative preference training with *Self-Refinement*-based inference. During training, EVOLVE enhances both direct question-answering performance and *Self-Refinement* capabilities. At inference time, it employs multi-step *Self-Refinement* to generate and refine outputs, creating preference data for subsequent training cycles. This synergistic loop leads to substantial improvements in LLM performance, surpassing GPT-4o on benchmarks such as AlpacaEval 2 and Arena-Hard. Furthermore, we investigate the potential of leveraging *Self-Refinement* to achieve *Self-Improvement* of the model’s intrinsic abilities. While this work primarily focuses on general tasks, future research will explore extending this framework to more complex reasoning tasks and broader domains, including mathematics and coding.

486 REFERENCES
487

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

491 Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to
492 retrieve, generate, and critique through self-reflection. 2024.

493

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

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

500

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

504 Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay
505 Srinivasan, Tianyi Zhou, Heng Huang, et al. Alpagasus: Training a better alpaca with fewer data.
506 *arXiv preprint arXiv:2307.08701*, 2023.

507

508 Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning
509 converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*,
510 2024.

511 Eugene Choi, Arash Ahmadian, Matthieu Geist, Olivier Pietquin, and Mohammad Gheshlaghi Azar.
512 Self-improving robust preference optimization. *arXiv preprint arXiv:2406.01660*, 2024.

513

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

517 Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser,
518 Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve
519 math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

520

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

523 Bosheng Ding, Chengwei Qin, Ruochen Zhao, Tianze Luo, Xinze Li, Guizhen Chen, Wenhan Xia,
524 Junjie Hu, Anh Tuan Luu, and Shafiq Joty. Data augmentation using large language models: Data
525 perspectives, learning paradigms and challenges. *arXiv preprint arXiv:2403.02990*, 2024.

526

527 Qingxiu Dong, Li Dong, Xingxing Zhang, Zhifang Sui, and Furu Wei. Self-boosting large language
528 models with synthetic preference data. *arXiv preprint arXiv:2410.06961*, 2024.

529

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

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

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

538

539 Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuxin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han.
Large language models can self-improve. *arXiv preprint arXiv:2210.11610*, 2022.

540 Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot,
 541 Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier,
 542 LÃ©lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril,
 543 Thomas Wang, Timothy Lacroix, and William El Sayed. Mistral 7b, 2023. URL <https://arxiv.org/abs/2310.06825>.

545 Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli,
 546 Shariq Iqbal, Colton Bishop, Rebecca Roelofs, et al. Training language models to self-correct via
 547 reinforcement learning. *arXiv preprint arXiv:2409.12917*, 2024.

548

549 Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-dpo: Step-
 550 wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*,
 551 2024.

552 Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman,
 553 Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in
 554 open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024a.

555

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

559

560 Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton
 561 Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, et al. Rlaif: Scaling reinforcement learning
 562 from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023.

563

564 Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E Gonzalez, and Ion
 565 Stoica. From live data to high-quality benchmarks: The arena-hard pipeline, 2024.

566

567 Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy
 568 Liang, and Tatsunori B Hashimoto. Alpacaeval: An automatic evaluator of instruction-following
 569 models, 2023.

570

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

574

575 Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang.
 576 On llms-driven synthetic data generation, curation, and evaluation: A survey. *arXiv preprint
 577 arXiv:2406.15126*, 2024.

578

579 Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri
 580 Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement
 581 with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2024.

582

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

585

586 Giovanni Monea, Antoine Bosselut, Kianté Brantley, and Yoav Artzi. Llms are in-context reinforce-
 587 ment learners. *arXiv preprint arXiv:2410.05362*, 2024.

588

589 Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong
 590 Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow
 591 instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:
 592 27730–27744, 2022.

593

594 Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West,
 595 and Boi Faltings. Refiner: Reasoning feedback on intermediate representations. *arXiv preprint
 596 arXiv:2304.01904*, 2023.

597

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

594 Snorkel. New benchmark results demonstrate value of snorkel ai approach to llm alignment, 2024.
 595 URL <https://snorkel.ai/new-benchmark-results-demonstrate-valueof>
 596 -snorkel-ai-approach-to-llm-alignment.

597 Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang.
 598 Preference ranking optimization for human alignment. *arXiv preprint arXiv:2306.17492*, 2023.

600 Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford,
 601 Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in*
 602 *neural information processing systems*, 33:3008–3021, 2020.

603 Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, Yongbin Li, Zhi Jin, Fei Huang,
 604 Dacheng Tao, and Jingren Zhou. A survey on self-evolution of large language models. *arXiv*
 605 *preprint arXiv:2404.14387*, 2024.

606 Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy
 607 Liang, and Tatsunori B Hashimoto. Alpaca: A strong, replicable instruction-following model.
 608 *Stanford Center for Research on Foundation Models*. <https://crfm.stanford.edu/2023/03/13/alpaca.html>, 3(6):7, 2023.

609 Ye Tian, Baolin Peng, Linfeng Song, Lifeng Jin, Dian Yu, Lei Han, Haitao Mi, and Dong Yu.
 610 Toward self-improvement of llms via imagination, searching, and criticizing. *Advances in Neural*
 611 *Information Processing Systems*, 37:52723–52748, 2024.

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

615 Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable preferences
 616 via multi-objective reward modeling and mixture-of-experts. *arXiv preprint arXiv:2406.12845*,
 617 2024.

618 Qibin Wang, Pu Zhao, Shaohan Huang, Fangkai Yang, Lu Wang, Furu Wei, Qingwei Lin, Saravan
 619 Rajmohan, and Dongmei Zhang. Learning to refine: Self-refinement of parallel reasoning in llms.
 620 *arXiv preprint arXiv:2509.00084*, 2025.

621 Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdh-
 622 ery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models.
 623 *arXiv preprint arXiv:2203.11171*, 2022a.

624 Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and
 625 Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions.
 626 *arXiv preprint arXiv:2212.10560*, 2022b.

627 Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang.
 628 Iterative preference learning from human feedback: Bridging theory and practice for rlhf under
 629 kl-constraint. *arXiv preprint arXiv:2312.11456*, 2023.

630 Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei
 631 Lin, and Dixin Jiang. Wizardlm: Empowering large pre-trained language models to follow complex
 632 instructions. In *The Twelfth International Conference on Learning Representations*, 2024.

633 Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen.
 634 Large language models as optimizers, 2024. URL <https://arxiv.org/abs/2309.03409>.

635 Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason
 636 Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

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

648 A ADDITIONAL RELATED WORKS
649650 **Reinforcement Learning from Human Feedback (RLHF).** RLHF has proven effective in aligning
651 LLMs with human values (Christiano et al., 2017; Ouyang et al., 2022; Bai et al., 2022a; Song et al.,
652 2023; Touvron et al., 2023). This approach uses human-annotated preference datasets to train a
653 Reward Model, guiding LLM optimization through reinforcement learning. However, due to the high
654 cost of human annotations, AI-generated feedback has been proposed to automate this process (Bai
655 et al., 2022b; Lee et al., 2023). Additionally, to reduce training costs, Direct Preference Optimization
656 (DPO) (Rafailov et al., 2023) bypasses the reward modeling process and directly aligns LLMs using
657 preference datasets. However, the effectiveness of these methods heavily depends on the quality of
658 the preference dataset, making the acquisition of high-quality preference data a critical challenge.
659660 **In-Context Learning (ICL).** ICL has become a fundamental capability of LLMs, enabling them to
661 perform tasks by conditioning on a few input examples without requiring parameter updates (Brown
662 et al., 2020). Recent studies, such as OPRO (Yang et al., 2024), show that LLMs can leverage their
663 ICL abilities to function as implicit optimizers, progressively improving performance on complex
664 problems. LLMs can also act as in-context reinforcement learners, optimizing behavior via reward
665 feedback (Monea et al., 2024). The SELF-REFINE (Madaan et al., 2024) is a special form of ICL.
666 It significantly enhances model performance through the FEEDBACK and REFINE mechanisms,
667 achieving remarkable results across multiple benchmarks. These findings indicate that integrating
668 ICL with model training presents a compelling strategy for constructing self-optimizing frameworks.
669670 B IMPLEMENTATION DETAILS OF BASELINES
671672 **Iterative DPO** (Snorkel, 2024; Xiong et al., 2023): We conduct the training process based on the
673 DPO *offline* model. Following the same online iterative training process as our method, we initially
674 extract a 5K prompt data from the UltraFeedback dataset (Cui et al., 2023) dataset. For each prompt,
675 we generate four responses in parallel. These responses are then scored and filtered using the Skywork
676 Reward Model (Liu et al., 2024) to construct a preference dataset, which is used to train the Iterative
677 DPO *iter1* model via DPO (Rafailov et al., 2023). Next, we extract a new 10K prompt data from the
678 UltraFeedback dataset and repeat the above process to obtain the Iterative DPO *iter2* model.
679680 **SynPO** (Dong et al., 2024): We utilize the 10K unused dataset from the UltraFeedback as Seed Data
681 to train the Response Improver. In each iteration, the Response Improver is trained using SFT, taking
682 a prompt x and the response y generated by the previous policy model M_{t-1} as input, with the chosen
683 response y_w from the Seed Data as the target output. For the policy model, we first generate responses
684 for the 5K prompt dataset from UltraFeedback using both the DPO *offline* model and the current
685 policy model M_{t-1} , yielding $\{y_{\text{off}}^{(i)}\}$ and $\{y^{(i)}\}$, respectively. The policy model's responses $\{y^{(i)}\}$
686 are then refined by the Response Improver to produce $\{\hat{y}^{(i)}\}$. We then use the Skywork Reward
687 Model to filter the $\{x^{(i)}, y_{\text{off}}^{(i)}, \hat{y}^{(i)}\}$ dataset, and train the DPO *offline* model to obtain the SynPO *iter1*
688 model. This process is repeated with a new 10K prompt dataset to train the SynPO *iter2* model.
689690 **SRPO** (Choi et al., 2024): We train the SRPO model starting from the SFT model, with two key
691 modifications to the Iterative DPO process. First, we replace the training algorithm with SRPO across
692 all stages. Second, as SRPO is an offline algorithm without a specified data collection process, we
693 adopt our *Self-Refinement* approach for data collection. The offline data used, including its size,
694 aligns with the training process of EVOLVE.
695696 **ScRe** (Kumar et al., 2024): Starting from the EVOLVE-SFT model, we use the first 45K prompts
697 from the UltraFeedback dataset and apply the REINFORCE algorithm for online training, following
698 ScRe's two-stage training paradigm.
699700 **Self-Rewarding** (Yuan et al., 2024): The training process of Self-Rewarding closely follows Iterative
701 DPO, generating four responses per prompt through parallel sampling and training the model using
702 DPO. The distinction lies in the construction of the preference dataset. Self-Rewarding employs
703 *LLM-as-a-Judge* to score each prompt-response pair (x, y) , assigning a score to response y . In our
704 experiments, the policy model itself serves as the *LLM-as-a-Judge*, evaluating the same dataset
705 over 3 rounds and taking the average score as the final quality assessment. The highest-scoring
706 response is selected as the chosen response and the lowest-scoring one as the rejected response. The
707 *LLM-as-a-Judge* template used aligns with the Self-Rewarding paper. Through two iterative training
708

702 rounds on the collected 5K and 10K preference datasets, we obtained the Self-Rewarding *iter1* and
 703 Self-Rewarding *iter2* models.

704 **SPIN** (Chen et al., 2024): The training process of SPIN is similar to Iterative DPO, with the key
 705 difference lying in the data collection method. For each prompt x , SPIN uses model to generate
 706 a response y , which is directly treated as the rejected response. The chosen response y_w from the
 707 preference dataset serves as the alignment target. Based on the DPO *offline* model, SPIN first collects
 708 preference data for 5K prompts from the UltraFeedback dataset and then conducts training to yield
 709 the SPIN *iter1* model. This process is then repeated with the SPIN *iter1* model and 10K prompt
 710 dataset from UltraFeedback, yielding the SPIN *iter2* model.

712 C MATHEMATICAL DERIVATIONS

714 C.1 THE OPTIMAL SOLUTION TO THE *Self-Refinement* OBJECTIVE IN THE PREFERENCE 715 TRAINING PHASE

716 In this Appendix, we aim to derive the loss function corresponding to the following objective:

$$719 \max_{\pi} \mathbb{E}_{y_2 \sim \pi(\cdot|x, y_1, z)} \left[p(y_2 \succ y_1|x) - \beta D_{\text{KL}}(\pi || \pi_{\text{ref}}|x, y_1, z) \right]. \quad (6)$$

721 First, we can obtain the optimal solution of the objective:

$$723 \max_{\pi} \mathbb{E}_{y_2 \sim \pi(\cdot|x, y_1, z)} \left[p(y_2 \succ y_1|x) - \beta D_{\text{KL}}(\pi || \pi_{\text{ref}}|x, y_1, z) \right] \quad (7)$$

$$725 = \max_{\pi} \mathbb{E}_{y_2 \sim \pi(\cdot|x, y_1, z)} \left[p(y_2 \succ y_1|x) - \beta \log \frac{\pi(y_2|x, y_1, z)}{\pi_{\text{ref}}(y_2|x, y_1, z)} \right] \quad (8)$$

$$728 = \max_{\pi} \beta \mathbb{E}_{y_2 \sim \pi(\cdot|x, y_1, z)} \left[-\log \frac{\pi(y_2|x, y_1, z)}{\pi_{\text{ref}}(y_2|x, y_1, z) \exp \left(\frac{p(y_2 \succ y_1|x)}{\beta} \right)} \right] \quad (9)$$

$$731 = \max_{\pi} -\beta \mathbb{E}_{y_2 \sim \pi(\cdot|x, y_1, z)} \left[\log \frac{\pi(y_2|x, y_1, z) Z(x, y_1, z)}{\pi_{\text{ref}}(y_2|x, y_1, z) \exp \left(\frac{p(y_2 \succ y_1|x)}{\beta} \right)} \right] + \beta \log Z(x, y_1, z) \quad (10)$$

$$734 = \max_{\pi} -\beta D_{\text{KL}} \left(\pi(y_2|x, y_1, z) \middle\| \frac{\pi_{\text{ref}}(y_2|x, y_1, z) \exp \left(\frac{p(y_2 \succ y_1|x)}{\beta} \right)}{Z(x, y_1, z)} \right) + \beta \log Z(x, y_1, z) \quad (11)$$

737 where $Z(x, y_1, z)$ is the partition function. Considering the non-negativity of the KL divergence, the
 738 optimal solution is:

$$740 \pi^*(y_2|x, y_1, z) = \frac{\pi_{\text{ref}}(y_2|x, y_1, z) \exp \left(\frac{p(y_2 \succ y_1|x)}{\beta} \right)}{Z(x, y_1, z)}. \quad (12)$$

743 Noting that $p(y_1 \succ y_1|x) = 1/2$, we derive the following expression:

$$745 \pi^*(y_1|x, y_1, z) = \frac{\pi_{\text{ref}}(y_1|x, y_1, z) \exp \left(\frac{1}{2\beta} \right)}{Z(x, y_1, z)}. \quad (13)$$

748 Dividing Eq.12 by Eq.13 yields

$$750 \frac{\pi^*(y_2|x, y_1, z)}{\pi^*(y_1|x, y_1, z)} = \frac{\pi_{\text{ref}}(y_2|x, y_1, z)}{\pi_{\text{ref}}(y_1|x, y_1, z)} \exp \left(\frac{p(y_2 \succ y_1|x)}{\beta} - \frac{1}{2\beta} \right). \quad (14)$$

753 Therefore, we have

$$755 \log \left(\frac{\pi^*(y_2|x, y_1, z) \pi_{\text{ref}}(y_1|x, y_1, z)}{\pi^*(y_1|x, y_1, z) \pi_{\text{ref}}(y_2|x, y_1, z)} \right) = \frac{p(y_2 \succ y_1|x)}{\beta} - \frac{1}{2\beta}. \quad (15)$$

756 By adopting the mean squared error as the loss function and parametrizing the policy model as π_θ ,
 757 we finally obtain:
 758

$$759 \mathcal{L}(\pi_\theta; \pi_{\text{ref}}) = \mathbb{E}_{(x, y_1, y_2) \sim \rho} \left[\log \left(\frac{\pi_\theta(y_2|x, y_1, z) \pi_{\text{ref}}(y_1|x, y_1, z)}{\pi_\theta(y_1|x, y_1, z) \pi_{\text{ref}}(y_2|x, y_1, z)} \right) - \left(\frac{p(y_2 \succ y_1|x)}{\beta} - \frac{1}{2\beta} \right) \right]^2, \quad (16)$$

762 where ρ represents the true distribution.
 763

764 D IMPLEMENTATION DETAILS

767 D.1 SELF-REFINEMENT TEMPLATE

769 The *Self-Refinement* template used in this paper is as follows:
 770

771 Self-Refinement Template

773 Below is a QUESTION from a user and an EXAMPLE RESPONSE.

774 Please provide a more helpful RESPONSE, improving the EXAMPLE RESPONSE by making the
 775 content even clearer, more accurate, and concise. Focus on addressing the human's QUESTION without
 776 including irrelevant sentences.

777 Your RESPONSE should not only be well-written, logical, and easy-to-follow, but also demonstrate
 778 expert-level insight, engaging the reader with the most relevant information.

779 **QUESTION:**
 780 {Question}

781 **EXAMPLE RESPONSE:**
 782 {Example_Response}

784 Now, refine and improve the RESPONSE further. You can consider two approaches:

785 1. **REFINEMENT:** If the EXAMPLE RESPONSE is sufficient and addresses most of the QUESTION's
 786 concerns, enhance clarity, accuracy, or conciseness as needed.
 787 2. **NEW RESPONSE:** If the EXAMPLE RESPONSE lacks clarity or relevance to the QUESTION,
 788 craft a more effective RESPONSE that thoroughly resolves the QUESTION.

789 Do not include analysis-just give the improved RESPONSE.

790 **RESPONSE:**

793 D.2 TRAINING DETAILS

795 In the SFT phase, we set the learning rate to 5×10^{-6} , with a batch size of 128 and a maximum
 796 sequence length of 1024. We employed a cosine learning rate schedule with 3% warm-up steps for 1
 797 epoch and used the AdamW optimizer.
 798

799 In the preference training phase, the learning rate was reduced to 1×10^{-6} . Additionally, we set
 800 $\alpha = 0.8$. For training with the Reward Model Scoring dataset filtering mechanism, we set $\beta = 0.01$,
 801 while for the Self-Improvement Rule-Based Selection mechanism, $\beta = 0.05$. The higher value of β
 802 in the Self-Improvement Rule-Based Selection process is due to the presence of noisy data in the
 803 filtered preference dataset, which requires stronger regularization.

805 D.3 INFERENCE DETAILS

807 During the iterative training and dataset collection process, we employed a sampling decoding strategy
 808 with a temperature of 0.7 for direct response generation and *Self-Refinement*. For AlpacaEval 2, we
 809 set the temperature to 0.9 for generation, while for MT-Bench and Arena-Hard, we followed the
 official decoding configuration. For GSM8K and MATH, we utilized a greedy decoding strategy.

810 D.4 EVALUATION BENCHMARKS DETAILS
811

812 AlpacaEval 2 (Li et al., 2023) consists of 805 questions from 5 datasets, MT-Bench (Zheng et al.,
813 2023) covers 80 questions across 8 categories in a multi-turn dialogue format, and Arena-Hard (Li
814 et al., 2024) is an enhanced version of MT-Bench with 500 well-defined technical questions. GSM8K
815 (Cobbe et al., 2021) includes elementary and middle school-level math problems, while MATH
816 (Hendrycks et al., 2021) contains more complex questions, spanning various mathematical branches
817 such as algebra, counting and probability, geometry, number theory, and calculus. We evaluate the
818 entire GSM8K test set, which contains 1319 math problems, and the first 1024 samples of the MATH
819 test set. For each benchmark, we report scores according to their respective evaluation protocols.
820

821 D.5 EXPERIMENT RESULTS ON MT-BENCH
822

823 For MT-Bench (Zheng et al., 2023), we report the average score using GPT-4 and GPT-4-Preview-
824 1106 as judges.
825

826 Table 4: Results on and MT-Bench (Zheng et al., 2023).
827

Method	MT-Bench	
	GPT-4 Turbo	GPT-4
SFT	6.4	6.9
+DPO <i>offline</i>	6.9	7.4
+Self-Rewarding <i>iter1</i>	6.9	7.5
+Self-Rewarding <i>iter2</i>	6.7	7.4
+Iterative DPO <i>iter1</i>	6.9	7.6
+Iterative DPO <i>iter2</i>	7.1	7.8
EVOLVE-SFT	6.4	7.0
+EVOLVE <i>offline</i>	7.0	7.6
+EVOLVE _{rule} <i>iter1</i>	7.1	7.5
+EVOLVE _{rule} <i>iter2</i>	7.1	7.7
+EVOLVE <i>iter1</i>	7.3	7.7
+EVOLVE <i>iter2</i>	7.7	8.1

841 D.6 COMPUTATIONAL OVERHEAD OF TRAINING AND INFERENCE
842

843 We compare the computational overhead between Iterative DPO and EVOLVE in the two-round
844 online iterative training, with the results presented as follows:
845

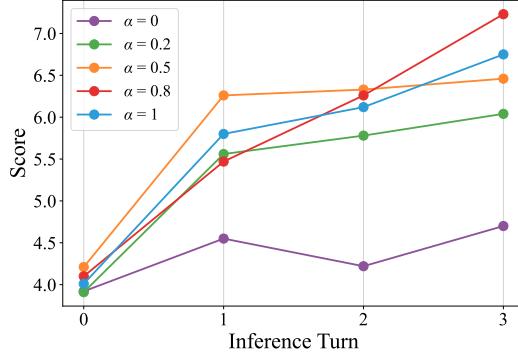
846 Table 5: Computational overhead during the training and inference stage.
847

Method	Iter1		Iter2	
	Dataset Collection (5k)	Training (5k)	Dataset Collection (10k)	Training (10k)
Iterative DPO	4h	25min	8h	50min
EVOLVE	5.5h	50min	11h	100min

855 To ensure a fair comparison, we introduce a normalization parameter α into our loss function, such
856 that the overall weight of the loss function during updates remains equal to 1.
857

859 D.7 ABLATION STUDY ON PARAMETER α
860

862 Fig. 10 shows the experimental results with different values of α on Llama-3.1-8B Base. Based on
863 the results which demonstrate the *Self-Refinement* capability performs optimally at $\alpha = 0.8$, we
selected $\alpha = 0.8$ in this work.
864

Figure 10: Experimental results with different values of α on Llama-3.1-8B Base.

E ALTERNATIVE REFINEMENT LOSS

E.1 THE DERIVATION OF THE REFINEMENT LOSS FUNCTION FROM THE BRADLEY-TERRY MODEL PERSPECTIVE

Alternatively, we can enhance the model’s *Self-Refinement* capability by leveraging insights from the Bradley-Terry (BT) model theory. We define the objective function as follows:

$$\max_{\pi} \mathbb{E}_{y_2 \sim \pi(\cdot|x, y_1, z)} \left[r(y_2|x, y_1, z) - \beta D_{\text{KL}}(\pi || \pi_{\text{ref}}|x, y_1, z) \right] \quad (17)$$

The solution process is analogous to that of Appendix C.1, allowing us to obtain the optimal solution:

$$\pi^*(y_2|x, y_1, z) = \frac{\pi_{\text{ref}}(y_2|x, y_1, z) \exp\left(\frac{r(y_2|x, y_1, z)}{\beta}\right)}{Z(x, y_1, z)}, \quad (18)$$

where $Z(x, y_1, z)$ is the partition function. Reorganizing the above equation, we obtain:

$$r(y_2|x, y_1, z) = \beta \log \frac{\pi(y_2|x, y_1, z)}{\pi_{\text{ref}}(y_2|x, y_1, z)} + \beta \log Z(x, y_1, z) \quad (19)$$

The standard expression of the BT model is:

$$p_{\text{BT}}^*(y_2 \succ y_1|x) = \sigma(r^*(y_2|x) - r^*(y_1|x)) \quad (20)$$

Here, to enhance the *Self-Refinement* capability of the language model, we make a slight modification. Given the problem input x for the BT model, we also provide an arbitrary response y_{opt} along with a refinement template z , which serves as guidance for the model to generate better responses:

$$p_{\text{BT}}^*(y_2 \succ y_1|x, y_{\text{opt}}, z) = \sigma(r^*(y_2|x, y_{\text{opt}}, z) - r^*(y_1|x, y_{\text{opt}}, z)). \quad (21)$$

Then we define the refinement preference function:

$$p_{\text{BT_refine}}^*(y_2 \succ y_1|x, z) = p_{\text{BT}}^*(y_2 \succ y_1|x, y_1, z) p_{\text{BT}}^*(y_2 \succ y_1|x, y_2, z) \quad (22)$$

$$= \sigma(r^*(y_2|x, y_1, z) - r^*(y_1|x, y_1, z)) \sigma(r^*(y_2|x, y_2, z) - r^*(y_1|x, y_2, z)) \quad (23)$$

$$\begin{aligned} &= \sigma \left(\beta \log \frac{\pi^*(y_2|x, y_1, z)}{\pi_{\text{ref}}(y_2|x, y_1, z)} - \beta \log \frac{\pi^*(y_1|x, y_1, z)}{\pi_{\text{ref}}(y_1|x, y_1, z)} \right) \\ &\quad \times \sigma \left(\beta \log \frac{\pi^*(y_2|x, y_2, z)}{\pi_{\text{ref}}(y_2|x, y_2, z)} - \beta \log \frac{\pi^*(y_1|x, y_2, z)}{\pi_{\text{ref}}(y_1|x, y_2, z)} \right) \end{aligned} \quad (24)$$

Assuming access to a well-curated preference dataset $\mathcal{D} = \{(x^{(i)}, y_w^{(i)}, y_l^{(i)})\}_{i=1}^N$, we aim to leverage this dataset to activate the *Self-Refinement* capability of language models, thereby gradually steering

the models toward generating better responses during the inference phase. To achieve this, we parametrize the policy model π_θ and estimate its parameters through maximum likelihood estimation. By treating the problem as a binary classification task, we have the negative log-likelihood loss:

$$\mathcal{L}_{\text{BT_SR}}(\pi) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w|x, y_l, z)}{\pi_{\text{ref}}(y_w|x, y_l, z)} - \beta \log \frac{\pi_\theta(y_l|x, y_l, z)}{\pi_{\text{ref}}(y_l|x, y_l, z)} \right) \right] - \mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_l|x, y_w, z)}{\pi_{\text{ref}}(y_l|x, y_w, z)} - \beta \log \frac{\pi_\theta(y_w|x, y_w, z)}{\pi_{\text{ref}}(y_w|x, y_w, z)} \right) \right] \quad (25)$$

Finally, we integrate the *Self-Refinement* loss with the DPO loss derived from the BT model perspective to obtain the EVOLVE loss function from the BT model viewpoint:

$$\mathcal{L}_{\text{BT_EVOLVE}}(\pi_\theta; \pi_{\text{ref}}) = \alpha \mathcal{L}_{\text{BT_SR}}(\pi_\theta; \pi_{\text{ref}}) + (1 - \alpha) \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}). \quad (26)$$

E.2 TRAINING DETAILS

During the SFT phase, BT_EVOLVE follows the same procedure as EVOLVE. The primary distinction between BT_EVOLVE and EVOLVE arises in the preference training phase, where we set $\beta = 0.05$ in BT_EVOLVE.

E.3 EXPERIMENT RESULTS

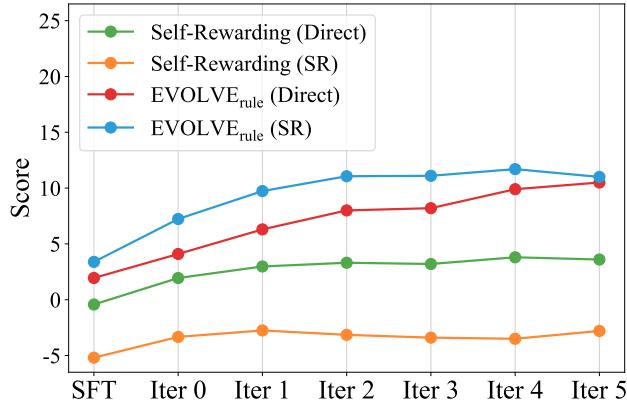
We compared BT_EVOLVE and EVOLVE across various benchmarks, with experimental results presented in Table 6. Both BT_EVOLVE and EVOLVE demonstrated nearly identical performance across all benchmarks, underscoring that the strength of our approach lies not in the algorithm itself, but in the foundational principles it embodies. Specifically, it is the concept of refinement that drives the effectiveness of our method and framework, enabling them to deliver impressive results.

Table 6: Comparison of experimental results between BT_EVOLVE and EVOLVE on AlpacaEval 2 (Li et al., 2023), Arena-Hard (Li et al., 2024), and MT-Bench (Zheng et al., 2023) under the **Direct** and **SR** settings. LC and WR represent length-controlled win rate and raw win rate, respectively.

Method	Direct					Self-Refinement (SR)		
	AlpacaEval 2		Arena-Hard		MT-Bench	AlpacaEval 2		Arena-Hard
	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4	LC (%)	WR (%)	WR (%)
BT_EVOLVE <i>offline</i>	19.8	19.3	20.0	7.0	7.5	27.8	25.7	24.8
BT_EVOLVE <i>iter1</i>	31.9	34.5	31.1	7.1	7.5	50.6	51.8	41.0
BT_EVOLVE <i>iter2</i>	45.2	47.7	39.5	7.4	7.7	66.2	66.6	49.9
EVOLVE <i>offline</i>	19.1	18.6	17.4	7.0	7.6	28.8	27.1	23.5
EVOLVE <i>iter1</i>	32.7	33.5	31.9	7.3	7.7	50.2	49.9	37.5
EVOLVE <i>iter2</i>	45.0	46.8	38.0	7.7	8.1	62.3	63.3	50.3

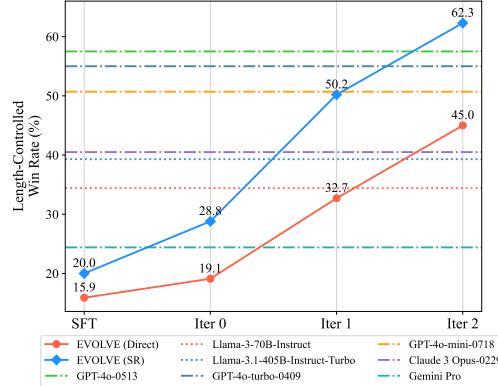
Table 7: Accuracy Comparison of BT_EVOLVE and EVOLVE on GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) tasks using **Direct** and **SR** generation strategies.

Method	GSM8K(%)		MATH(%)	
	Direct	SR	Direct	SR
BT_EVOLVE <i>offline</i>	67.6	71.7	37.7	44.1
BT_EVOLVE <i>iter1</i>	70.3	73.2	46.5	47.4
BT_EVOLVE <i>iter2</i>	70.1	71.6	50.2	52.0
EVOLVE <i>offline</i>	68.3	72.2	38.3	42.4
EVOLVE <i>iter1</i>	70.2	71.7	46.1	47.8
EVOLVE <i>iter2</i>	71.9	73.6	48.7	50.1

972 F FURTHER EXPLORATION OF SELF-IMPROVEMENT WITH MORE ITERATIONS
973
974

988 Figure 11: Investigation of the upper bound of the training iterations. We divided the last 15K samples
989 from the UltraFeedback dataset into 3 subsets (5K each) and performed three additional training
990 iterations. The evaluations are performed on the UltraFeedback test dataset.

991 As shown in Fig. 11, EVOLVE_{rule}'s performance of *Self-Refinement (SR)* has essentially reached its
992 upper bound, exhibiting stabilization with minor fluctuation within a certain range. Meanwhile,
993 the performance of Direct Response Generation (**Direct**) continues to improve. Based on this trend,
994 the performance of **Direct** is expected to approach that of SR, achieving little or almost no gap. Besides,
995 the Self-Rewarding demonstrates performance convergence by Iter 3 according to the trend.
996

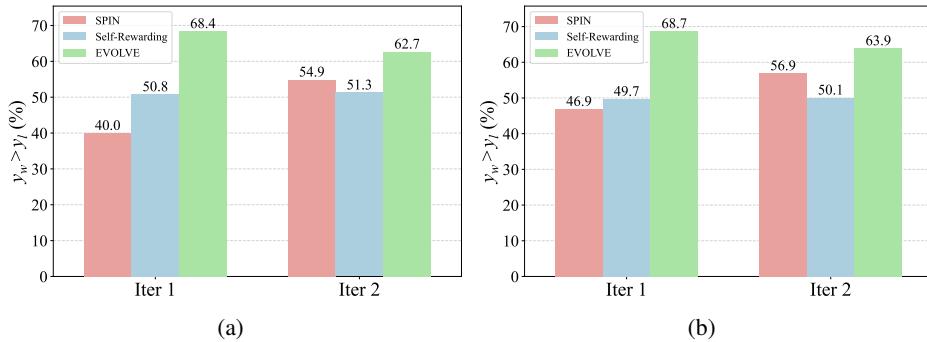
997 G COMPARISON WITH OPEN-SOURCE MODELS
998

1000 Figure 12: Length-controlled win rate on AlpacaEval 2 improves with EVOLVE iterations,
1001 surpassing GPT-4 level for the base versions of
1002 Llama-3.1-8B when utilizing the *Self-Refinement*
1003 strategy.

Model	Size	LC(%)	WR(%)
Llama-3.1-8B-Base-EVOLVE (iter2, SR)	8B	62.3	63.3
GPT-4o (05/13)	-	57.5	51.3
GPT-4-turbo (04/09)	-	55.0	46.1
GPT-4o-mini (07/18)	-	50.7	44.7
Llama-3.1-8B-Base-EVOLVE (iter1, SR)	8B	50.2	49.9
GPT-4_1106_preview	-	50.0	50.0
Llama-3.1-8B-Base-EVOLVE (iter2, Direct)	8B	45.0	46.8
Claude 3 Opus (02/29)	-	40.5	29.1
Llama-3.1-405B-Instruct-Turbo	405B	39.3	39.1
Qwen2-72B-Instruct	72B	38.1	29.9
Llama-3-70B-Instruct	70B	34.4	33.2
Llama-3.1-8B-Base-EVOLVE (iter1, Direct)	8B	32.7	33.5
Mistral Large (24/02)	123B	32.7	21.4
Gemini Pro	-	24.4	18.2
Llama-3.1-8B-Instruct	8B	20.9	21.8

1004 Table 8: Results on AlpacaEval 2 leaderboard.
1005 LC and WR represent length-controlled and raw
1006 win rate, respectively. "Direct" refers to the di-
1007 rect response generation strategy, while "SR"
1008 denotes the *Self-Refinement* generation strategy.

1026
1027 **H QUALITY EVALUATION OF DATA COLLECTED UNDER THE**
1028 **SELF-IMPROVEMENT SETTING**



1041
1042 Figure 13: Quality Evaluation of the Dataset Collected under the Self-Improvement Setting. We
1043 evaluate the dataset collected without supervised signals using two reward models: Skywork Reward
1044 Model (Liu et al., 2024) (Figure a) and ArmoRM (Wang et al., 2024) (Figure b). Both reward
1045 models are 8B Parameter Scale and have demonstrated outstanding performance on the RewardBench
1046 leaderboard (Lambert et al., 2024b). The y-axis indicates the percentage of samples in the dataset for
1047 which the chosen response achieves a higher reward score compared to the rejected response. By
1048 leveraging different reward models for scoring, we aim to minimize potential biases in the evaluation
1049 results. The assessments from both reward models consistently demonstrate the effectiveness of the
1050 Self-Refinement generation strategy in enhancing the dataset quality.

1051 **I DETAILS OF ABLATION STUDIES**

1052 **I.1 DPO LOSS AND GENERATION STRATEGY**

1053 The specific experimental results for Fig. 7 and Fig. 8 are presented in Tab. 9 below.

1054 Table 9: Ablation study on AlpacaEval 2 and Arena-Hard. LC and WR represent length-controlled
1055 win rate and raw win rate, respectively.

Method	Direct			Self-Refinement (SR)		
	AlpacaEval 2		Arena-Hard	AlpacaEval 2		Arena-Hard
	LC (%)	WR (%)	WR (%)	LC (%)	WR (%)	WR (%)
EVOLVE <i>iter1</i> using PS	28.7	29.7	29.8	41.0	42.5	35.5
EVOLVE <i>iter2</i> using PS	40.7	42.7	37.1	59.4	60.6	49.6
EVOLVE <i>offline</i> w/o DPO	19.3	19.2	18.3	25.9	24.8	25.0
EVOLVE <i>iter1</i> w/o DPO	31.4	33.0	29.6	47.4	49.2	39.5
EVOLVE <i>iter2</i> w/o DPO	38.3	41.3	36.0	54.9	56.3	48.6
EVOLVE <i>offline</i>	19.1	18.6	17.4	28.8	27.1	23.5
EVOLVE <i>iter1</i>	32.7	33.5	31.9	50.2	49.9	37.5
EVOLVE <i>iter2</i>	45.0	46.8	38.0	62.3	63.3	50.3

1073 **I.2 IMPACT OF DIFFERENT REWARD MODELS**

1074 We also conducted ablation studies on the influence of different reward models adopted during
1075 the entire iterative training process. The results are presented in Tab. 10. It can be seen that the
1076 performance of the models trained with ArmoRM is generally consistent with that of the models
1077 trained with the Skywork Reward Model reported in this paper. This suggests that the effectiveness
1078 of our framework is not dependent on the reward model, and its ability to continually enhance the
1079 Self-Refinement capability is robust to this choice.

1080 Table 10: Results on AlpacaEval 2 (Li et al., 2023) and Arena-Hard (Li et al., 2024) under the **Direct**
 1081 and **SR** settings, when using **Skywork Reward Model** and **ArmoRM** as the reward model to retrain
 1082 the entire iterative process based on the EVOLVE *offline* model, respectively.

Method	Direct			Self-Refinement (SR)		
	AlpacaEval 2		Arena-Hard	AlpacaEval 2		Arena-Hard
	LC (%)	WR (%)	WR (%)	LC (%)	WR (%)	WR (%)
SFT	15.9	12.7	12.7	13.8	8.1	8.0
EVOLVE-SFT	15.9	15.5	16.5	20.0	18.6	18.0
EVOLVE <i>offline</i>	19.1	18.6	17.4	28.8	27.1	23.5

Skywork Reward Model						
EVOLVE <i>iter1</i>	32.7	33.5	31.9	50.2	49.9	37.5
EVOLVE <i>iter2</i>	<u>45.0</u>	46.8	<u>38.0</u>	<u>62.3</u>	63.3	<u>50.3</u>

ArmoRM						
EVOLVE <i>iter1</i>	34.1	28.8	29.3	48.7	35.7	39.1
EVOLVE <i>iter2</i>	45.4	<u>42.5</u>	40.6	63.8	<u>58.1</u>	56.4

1102 J EVALUATION OF SELF-REFINEMENT CAPABILITIES IN OPEN-SOURCE 1103 MODELS

1104
 1105
 1106
 1107 In this section, we evaluate the *Self-Refinement* capabilities of several prominent open-source models.
 1108 To provide a comprehensive assessment, we examine the performance of the Llama-3.1-8B-Instruct,
 1109 Qwen2.5-7B-Instruct, and Gemma-2-9B-Instruct models across three different refinement templates.
 1110 The templates employed in this evaluation are as follows:

1112 Direct Refinement Template

1113 Below is a QUESTION from a user and an EXAMPLE RESPONSE.
 1114 Please provide a more helpful RESPONSE, improving the EXAMPLE RESPONSE by making the
 1115 content even clearer, more accurate, and concise. Focus on addressing the human's QUESTION without
 1116 including irrelevant sentences.
 1117 Your RESPONSE should not only be well-written, logical, and easy-to-follow, but also demonstrate
 1118 expert-level insight, engaging the reader with the most relevant information.

1119
 1120
 1121
 1122
 1123
 1124
 1125
 1126
 1127
 1128
 1129
 1130
 1131
 1132
 1133
 1134
 1135
 1136
 1137
 1138
 1139
 1140
 1141
 1142
 1143
 1144
 1145
 1146
 1147
 1148
 1149
 1150
 1151
 1152
 1153
 1154
 1155
 1156
 1157
 1158
 1159
 1160
 1161
 1162
 1163
 1164
 1165
 1166
 1167
 1168
 1169
 1170
 1171
 1172
 1173
 1174
 1175
 1176
 1177
 1178
 1179
 1180
 1181
 1182
 1183
 1184
 1185
 1186
 1187
 1188
 1189
 1190
 1191
 1192
 1193
 1194
 1195
 1196
 1197
 1198
 1199
 1200
 1201
 1202
 1203
 1204
 1205
 1206
 1207
 1208
 1209
 1210
 1211
 1212
 1213
 1214
 1215
 1216
 1217
 1218
 1219
 1220
 1221
 1222
 1223
 1224
 1225
 1226
 1227
 1228
 1229
 1230
 1231
 1232
 1233
 1234
 1235
 1236
 1237
 1238
 1239
 1240
 1241
 1242
 1243
 1244
 1245
 1246
 1247
 1248
 1249
 1250
 1251
 1252
 1253
 1254
 1255
 1256
 1257
 1258
 1259
 1260
 1261
 1262
 1263
 1264
 1265
 1266
 1267
 1268
 1269
 1270
 1271
 1272
 1273
 1274
 1275
 1276
 1277
 1278
 1279
 1280
 1281
 1282
 1283
 1284
 1285
 1286
 1287
 1288
 1289
 1290
 1291
 1292
 1293
 1294
 1295
 1296
 1297
 1298
 1299
 1300
 1301
 1302
 1303
 1304
 1305
 1306
 1307
 1308
 1309
 1310
 1311
 1312
 1313
 1314
 1315
 1316
 1317
 1318
 1319
 1320
 1321
 1322
 1323
 1324
 1325
 1326
 1327
 1328
 1329
 1330
 1331
 1332
 1333
 1334
 1335
 1336
 1337
 1338
 1339
 1340
 1341
 1342
 1343
 1344
 1345
 1346
 1347
 1348
 1349
 1350
 1351
 1352
 1353
 1354
 1355
 1356
 1357
 1358
 1359
 1360
 1361
 1362
 1363
 1364
 1365
 1366
 1367
 1368
 1369
 1370
 1371
 1372
 1373
 1374
 1375
 1376
 1377
 1378
 1379
 1380
 1381
 1382
 1383
 1384
 1385
 1386
 1387
 1388
 1389
 1390
 1391
 1392
 1393
 1394
 1395
 1396
 1397
 1398
 1399
 1400
 1401
 1402
 1403
 1404
 1405
 1406
 1407
 1408
 1409
 1410
 1411
 1412
 1413
 1414
 1415
 1416
 1417
 1418
 1419
 1420
 1421
 1422
 1423
 1424
 1425
 1426
 1427
 1428
 1429
 1430
 1431
 1432
 1433
 1434
 1435
 1436
 1437
 1438
 1439
 1440
 1441
 1442
 1443
 1444
 1445
 1446
 1447
 1448
 1449
 1450
 1451
 1452
 1453
 1454
 1455
 1456
 1457
 1458
 1459
 1460
 1461
 1462
 1463
 1464
 1465
 1466
 1467
 1468
 1469
 1470
 1471
 1472
 1473
 1474
 1475
 1476
 1477
 1478
 1479
 1480
 1481
 1482
 1483
 1484
 1485
 1486
 1487
 1488
 1489
 1490
 1491
 1492
 1493
 1494
 1495
 1496
 1497
 1498
 1499
 1500
 1501
 1502
 1503
 1504
 1505
 1506
 1507
 1508
 1509
 1510
 1511
 1512
 1513
 1514
 1515
 1516
 1517
 1518
 1519
 1520
 1521
 1522
 1523
 1524
 1525
 1526
 1527
 1528
 1529
 1530
 1531
 1532
 1533
 1534
 1535
 1536
 1537
 1538
 1539
 1540
 1541
 1542
 1543
 1544
 1545
 1546
 1547
 1548
 1549
 1550
 1551
 1552
 1553
 1554
 1555
 1556
 1557
 1558
 1559
 1560
 1561
 1562
 1563
 1564
 1565
 1566
 1567
 1568
 1569
 1570
 1571
 1572
 1573
 1574
 1575
 1576
 1577
 1578
 1579
 1580
 1581
 1582
 1583
 1584
 1585
 1586
 1587
 1588
 1589
 1590
 1591
 1592
 1593
 1594
 1595
 1596
 1597
 1598
 1599
 1600
 1601
 1602
 1603
 1604
 1605
 1606
 1607
 1608
 1609
 1610
 1611
 1612
 1613
 1614
 1615
 1616
 1617
 1618
 1619
 1620
 1621
 1622
 1623
 1624
 1625
 1626
 1627
 1628
 1629
 1630
 1631
 1632
 1633
 1634
 1635
 1636
 1637
 1638
 1639
 1640
 1641
 1642
 1643
 1644
 1645
 1646
 1647
 1648
 1649
 1650
 1651
 1652
 1653
 1654
 1655
 1656
 1657
 1658
 1659
 1660
 1661
 1662
 1663
 1664
 1665
 1666
 1667
 1668
 1669
 1670
 1671
 1672
 1673
 1674
 1675
 1676
 1677
 1678
 1679
 1680
 1681
 1682
 1683
 1684
 1685
 1686
 1687
 1688
 1689
 1690
 1691
 1692
 1693
 1694
 1695
 1696
 1697
 1698
 1699
 1700
 1701
 1702
 1703
 1704
 1705
 1706
 1707
 1708
 1709
 1710
 1711
 1712
 1713
 1714
 1715
 1716
 1717
 1718
 1719
 1720
 1721
 1722
 1723
 1724
 1725
 1726
 1727
 1728
 1729
 1730
 1731
 1732
 1733
 1734
 1735
 1736
 1737
 1738
 1739
 1740
 1741
 1742
 1743
 1744
 1745
 1746
 1747
 1748
 1749
 1750
 1751
 1752
 1753
 1754
 1755
 1756
 1757
 1758
 1759
 1760
 1761
 1762
 1763
 1764
 1765
 1766
 1767
 1768
 1769
 1770
 1771
 1772
 1773
 1774
 1775
 1776
 1777
 1778
 1779
 1780
 1781
 1782
 1783
 1784
 1785
 1786
 1787
 1788
 1789
 1790
 1791
 1792
 1793
 1794
 1795
 1796
 1797
 1798
 1799
 1800
 1801
 1802
 1803
 1804
 1805
 1806
 1807
 1808
 1809
 1810
 1811
 1812
 1813
 1814
 1815
 1816
 1817
 1818
 1819
 1820
 1821
 1822
 1823
 1824
 1825
 1826
 1827
 1828
 1829
 1830
 1831
 1832
 1833
 1834
 1835
 1836
 1837
 1838
 1839
 1840
 1841
 1842
 1843
 1844
 1845
 1846
 1847
 1848
 1849
 1850
 1851
 1852
 1853
 1854
 1855
 1856
 1857
 1858
 1859
 1860
 1861
 1862
 1863
 1864
 1865
 1866
 1867
 1868
 1869
 1870
 1871
 1872
 1873
 1874
 1875
 1876
 1877
 1878
 1879
 1880
 1881
 1882
 1883
 1884
 1885
 1886
 1887
 1888
 1889
 1890
 1891
 1892
 1893
 1894
 1895
 1896
 1897
 1898
 1899
 1900
 1901
 1902
 1903
 1904
 1905
 1906
 1907
 1908
 1909
 1910
 1911
 1912
 1913
 1914
 1915
 1916
 1917
 1918
 1919
 1920
 1921
 1922
 1923
 1924
 1925
 1926
 1927
 1928
 1929
 1930
 1931
 1932
 1933
 1934
 1935
 1936
 1937
 1938
 1939
 1940
 1941
 1942
 1943
 1944
 1945
 1946
 1947
 1948
 1949
 1950
 1951
 1952
 1953
 1954
 1955
 1956
 1957
 1958
 1959
 1960
 1961
 1962
 1963
 1964
 1965
 1966
 1967
 1968
 1969
 1970
 1971
 1972
 1973
 1974
 1975
 1976
 1977
 1978
 1979
 1980
 1981
 1982
 1983
 1984
 1985
 1986
 1987
 1988
 1989
 1990
 1991
 1992
 1993
 1994
 1995
 1996
 1997
 1998
 1999
 2000
 2001
 2002
 2003
 2004
 2005
 2006
 2007
 2008
 2009
 2010
 2011
 2012
 2013
 2014
 2015
 2016
 2017
 2018
 2019
 2020
 2021
 2022
 2023
 2024
 2025
 2026
 2027
 2028
 2029
 2030
 2031
 2032
 2033
 2034
 2035
 2036
 2037
 2038
 2039
 2040
 2041
 2042
 2043
 2044
 2045
 2046
 2047
 2048
 2049
 2050
 2051
 2052
 2053
 2054
 2055
 2056
 2057
 2058
 2059
 2060
 2061
 2062
 2063
 2064
 2065
 2066
 2067
 2068
 2069
 2070
 2071
 2072
 2073
 2074
 2075
 2076
 2077
 2078
 2079
 2080
 2081
 2082
 2083
 2084
 2085
 2086
 2087
 2088
 2089
 2090
 2091
 2092
 2093
 2094
 2095
 2096
 2097
 2098
 2099
 2100
 2101
 2102
 2103
 2104
 2105
 2106
 2107
 2108
 2109
 2110
 2111
 2112
 2113
 2114
 2115
 2116
 2117
 2118
 2119
 2120
 2121
 2122
 2123
 2124
 2125
 2126
 2127
 2128
 2129
 2130
 2131
 2132
 2133
 2134
 2135
 2136
 2137
 2138
 2139
 2140
 2141
 2142
 2143
 2144
 2145
 2146
 2147
 2148
 2149
 2150
 2151
 2152
 2153
 2154
 2155
 2156
 2157
 2158
 2159
 2160
 2161
 2162
 2163
 2164
 2165
 2166
 2167
 2168
 2169
 2170
 2171
 2172
 2173
 2174
 2175
 2176
 2177
 2178
 2179
 2180
 2181
 2182
 2183
 2184
 2185
 2186
 2187
 2188
 2189
 2190
 2191
 2192
 2193
 2194
 2195
 2196
 2197
 2198
 2199
 2200
 2201
 2202
 2203
 2204
 2205
 2206
 2207
 2208
 2209
 2210
 2211
 2212
 2213
 2214
 2215
 2216
 2217
 2218
 2219
 2220
 2221
 2222
 2223
 2224
 2225
 2226
 2227
 2228
 2229
 2230
 2231
 2232
 2233
 2234
 2235
 2236
 2237
 2238
 2239
 2240
 2241
 2242
 2243
 2244
 2245
 2246
 2247
 2248
 2249
 2250
 2251
 2252
 2253
 2254
 2255
 2256
 2257
 2258
 2259
 2260
 226

1134
1135

Analysis-Guided Refinement Template

1136
1137
1138
1139
1140
1141

Below is a QUESTION from a user and an EXAMPLE RESPONSE.

Please provide a more helpful RESPONSE, improving the EXAMPLE RESPONSE by making the content even clearer, more accurate, and concise. Focus on addressing the human's QUESTION without including irrelevant sentences.

Your RESPONSE should not only be well-written, logical, and easy-to-follow, but also demonstrate expert-level insight, engaging the reader with the most relevant information.

1142
1143
1144
1145
1146QUESTION:
{Question}EXAMPLE RESPONSE:
{Example_Response}1147
1148
1149
1150
1151

Now, refine and improve the RESPONSE further. You can consider two approaches:

1. REFINEMENT: If the EXAMPLE RESPONSE is sufficient and addresses most of the QUESTION's concerns, enhance clarity, accuracy, or conciseness as needed.
2. NEW RESPONSE: If the EXAMPLE RESPONSE lacks clarity or relevance to the QUESTION, craft a more effective RESPONSE that thoroughly resolves the QUESTION.

1152
1153
1154
1155
1156

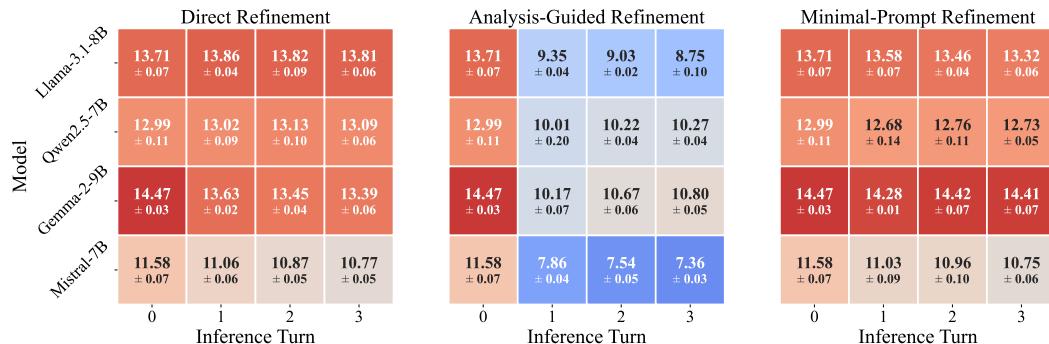
Format your answer as follows:

ANALYSIS: <Analyze the strengths and shortcomings of the EXAMPLE RESPONSE>
RESPONSE: <Provide an improved response>1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170

Minimal-Prompt Refinement Template

Below is a QUESTION from a user and an EXAMPLE RESPONSE.
Please provide a better RESPONSE.QUESTION:
{Question}EXAMPLE RESPONSE:
{Example_Response}

RESPONSE:

1182
1183
1184
1185
1186
1187Figure 14: Evaluation of *Self-Refinement* Capability Across Various Models. We use three refinement templates to minimize prompt bias. The x-axis denotes the inference iteration number. For each turn, responses are generated from 256 UltraFeedback test set samples, using the original prompt and the prior turn's output. These are then scored by the ArmoRM (Wang et al., 2024). To eliminate potential randomness, the reported values are the mean score of three independent runs with different random seeds. For better visualization, ArmoRM scores are scaled by a factor of 100 due to their originally narrow value range; higher scores indicate better quality.

1188 To ensure robust evaluation and mitigate biases from a single reward model, we employed two
1189 distinct reward models—Skywork Reward Model (Liu et al., 2024) and ArmoRM (Wang et al.,
1190 2024)—to score the results, as shown in Fig. 1 (Skywork Reward Model) and Fig. 14 (ArmoRM). Our
1191 experiments demonstrate that, despite their widespread adoption, current open-source models often
1192 struggle to effectively refine their responses, which can even lead to a degradation in performance.
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241

1242 **K ANALYSIS OF VARIOUS DYNAMIC GENERATION STRATEGIES**
12431244 **Prompt Template of Chain of Self-Refinement Generation Strategy:**
12451246 **Chain of Self-Refinement Template**
12471248 Below is a QUESTION from a user and an EXAMPLE RESPONSE.
12491250 Please provide a more helpful RESPONSE, improving the EXAMPLE RESPONSE by making the
1251 content even clearer, more accurate, and concise. Focus on addressing the human's QUESTION without
1252 including irrelevant sentences.
12531254 Your RESPONSE should not only be well-written, logical, and easy-to-follow, but also demonstrate
1255 expert-level insight, engaging the reader with the most relevant information.
12561257 **QUESTION:**
1258 {Question}1259 **EXAMPLE RESPONSE:**
1260 {Example_Response}1261 Now, refine and improve the RESPONSE further. You can consider two approaches:
12621. **REFINEMENT:** If the EXAMPLE RESPONSE is sufficient and addresses most of the QUESTION's
1263 concerns, enhance clarity, accuracy, or conciseness as needed.
2. **NEW RESPONSE:** If the EXAMPLE RESPONSE lacks clarity or relevance to the QUESTION,
1264 craft a more effective RESPONSE that thoroughly resolves the QUESTION.
12651266 Do not include analysis-just give the improved RESPONSE.
12671268 **RESPONSE:**
12691270 **Prompt Template of Few-Shot Self-Refinement Generation Strategy:**
12711272 **Few-Shot Self-Refinement Template**
12731274 Below is a QUESTION from a user and several EXAMPLE RESPONSES, ordered in a *Self-Refinement*
1275 sequence.
12761277 Please provide a more helpful RESPONSE, improving the previous EXAMPLE RESPONSES by
1278 making the content even clearer, more accurate, and concise. Focus on addressing the human's
1279 QUESTION without including irrelevant sentences.
12801281 Your RESPONSE should not only be well-written, logical, and easy-to-follow, but also demonstrate
1282 expert-level insight, engaging the reader with the most relevant information.
12831284 **QUESTION:**
1285 {Question}1286 **EXAMPLE RESPONSES (in *Self-Refinement* order):**
1287 {Example_Responses}1288 Now, refine and improve the RESPONSE further. You can consider two approaches:
12891. **REFINEMENT:** If the EXAMPLE RESPONSES are sufficient and addresses most of the QUES-
1290 TION's concerns, enhance clarity, accuracy, or conciseness as needed.
2. **NEW RESPONSE:** If the EXAMPLE RESPONSES lack clarity or relevance to the QUESTION,
1291 craft a more effective RESPONSE that thoroughly resolves the QUESTION.
12921293 Do not include analysis-just give the improved RESPONSE.
12941295 **RESPONSE:**
12961297 **Self-Refinement with Self-Evaluation Generation Strategy:** The Self-Refinement with Self-
1298 Evaluation generation strategy adopts the same Self-Refinement template as the Chain of Self-
1299 Refinement generation strategy, thereby minimizing potential performance degradation caused by
1295 template variations.

1296
1297

Self-Evaluation Template

1298
1299

Below is a QUESTION from a user and a RESPONSE provided by an AI system.

1300
1301

QUESTION:

{Question}

1302
1303

RESPONSE:

{Example_Response}

1304
1305
1306

Please provide a detailed and comprehensive ANALYSIS of both the QUESTION and the RESPONSE. Your ANALYSIS should focus on the following aspects:

1307
1308
1309

1. Understanding the Question: Evaluate how well the AI system understood the intent and nuances of the QUESTION. Highlight any gaps or misinterpretations, and suggest ways to better align with the user's needs.

1310
1311

2. Quality of the Response:

1312
1313

- Accuracy: Assess whether the RESPONSE is factually correct and provides relevant information. Point out any inaccuracies or missing details.
- Depth: Analyze whether the RESPONSE is sufficiently detailed and addresses the QUESTION fully. Suggest areas where additional depth or examples could enhance the RESPONSE.
- Clarity: Comment on the RESPONSE's clarity, structure, and language. Identify any ambiguous or confusing parts and recommend improvements.
- Engagement: Evaluate how engaging and conversational the RESPONSE is, considering tone and style. Suggest ways to make it more user-friendly and engaging.
- Safety: Ensure the RESPONSE avoids offensive, harmful, toxic, or inappropriate content. Assess whether it adheres to ethical and safety guidelines, promoting inclusivity and respect. Highlight any instances of potentially harmful or inappropriate content, and suggest strategies for improvement.

1314
1315

3. Suggestions for Improvement:

1316
1317

- Provide actionable recommendations to improve the RESPONSE quality.
- Suggest alternative approaches or strategies the AI system could use to deliver a better RESPONSE in the future.

1318
1319

Your ANALYSIS must be factual, insightful, and constructive, with a clear focus on enhancing the quality, usefulness, and safety of future RESPONSES. Ensure that all points are rooted in verifiable facts or evidence, avoiding any fabrication or speculative statements. Keep your analysis concise, clear, and to the point, avoiding unnecessary elaboration.

1320
1321
1322

ANALYSIS:

1323
13241325
1326

1327

1328
1329
13301331
1332
13331334
13351336
13371338
13391340
13411342
13431344
13451346
13471348
1349

1350 Examples of Self-Evaluation:
 1351
 1352

1353 **Prompt:**

1354 Gloria is shoe shopping when she comes across a pair of boots that fit her shoe budget. However, she
 1355 has to choose between the boots and two pairs of high heels that together cost five dollars less than
 1356 the boots. If one pair of heels costs \$33 and the other costs twice as much, how many dollars are the
 1357 boots?

1358 **Example Response:**

1359 Let's break down the problem step by step:

1360 1. One pair of high heels costs \$33.
 1361 2. The other pair of high heels costs twice as much as the first pair, so it costs $33 * 2 = 66$ dollars.
 1362 3. The total cost of the two pairs of high heels is $33 + 66 = 99$ dollars.
 1363 4. The boots cost five dollars less than the two pairs of high heels, so the boots cost $99 - 5 = 94$ dollars.

1364
 1365 Therefore, the boots cost \$94.

1366 **EVOLVE Self-Evaluation:**

1367 1. Understanding the Question: The AI system accurately understood the intent and nuances of the
 1368 QUESTION. The problem is well-defined, and the AI system correctly identified the key elements,
 1369 such as the cost of the boots, the two pairs of high heels, and the relationship between their costs.

1370
 1371 2. Quality of the Response:

1372 - Accuracy: The RESPONSE is factually correct and provides relevant information. The AI
 1373 system correctly calculated the cost of the boots based on the given information.
 1374 - Depth: The RESPONSE is sufficiently detailed and addresses the QUESTION fully. The AI
 1375 system provided a clear and concise explanation of the solution, including the steps taken to
 1376 arrive at the answer.
 1377 - Clarity: The RESPONSE is clear, structured, and well-written. The AI system used appro-
 1378 priate language and tone, making the response easy to understand.
 1379 - Engagement: The RESPONSE is engaging and conversational. The AI system used a
 1380 friendly and approachable tone, making the response more user-friendly.
 1381 - Safety: The RESPONSE avoids offensive, harmful, toxic, or inappropriate content. The AI
 1382 system adheres to ethical and safety guidelines, promoting inclusivity and respect.

1383
 1384 3. Suggestions for Improvement:

1385 - Provide a more detailed explanation of the solution, including the reasoning behind each
 1386 step. This would help users better understand the problem-solving process.
 1387 - Offer alternative approaches or strategies, such as using a different method to solve the
 1388 problem or providing a visual representation of the solution.
 1389 - Consider incorporating more examples or scenarios to illustrate the problem-solving process
 1390 and demonstrate the AI system's capabilities.

1391 In conclusion, the AI system provided a high-quality RESPONSE that accurately addressed the
 1392 QUESTION. The response was factually correct, sufficiently detailed, clear, engaging, and safe.
 1393 However, there is room for improvement, particularly in providing a more detailed explanation of the
 1394 solution and offering alternative approaches or strategies. By incorporating these suggestions, the AI
 1395 system can deliver an even better RESPONSE in the future.

1396
 1397 Table 11: An Example of Model Self-Evaluation. The small language model without domain-specific
 1398 evaluation training demonstrates limited self-evaluation capability. Most of its assessments tend to be
 1399 superficial or overly accommodating, often aligning uncritically with the provided answer's reasoning.
 1400 In this particular case, the given solution incorrectly interpreted the problem's requirements in its
 1401 final step, leading to an erroneous conclusion (**the correct answer should be $99 + 5 = 104$ dollars**).
 1402 However, the model fails to constructively identify this critical error.

1403

1404

Prompt:

1405

Raymond and Samantha are cousins. Raymond was born 6 years before Samantha. Raymond had a son at the age of 23. If Samantha is now 31, how many years ago was Raymond's son born?

1406

1407

Example Response:

1408

1409

Samantha is currently 31 years old. Raymond was born 6 years before Samantha, so Raymond is currently 25 years old. Raymond had his son when he was 23 years old. To find out how many years ago Raymond's son was born, we subtract Raymond's age when his son was born from his current age. So, 25 (Raymond's current age) - 23 (Raymond's age when his son was born) = 2 years ago. Therefore, Raymond's son was born 2 years ago.

1410

1411

1412

1413

1414

The answer is: 2 years ago.

1415

EVOLVE Self-Evaluation:

1416

1417

1418

1419

1420

1421

1422

2. Quality of the Response:

1423

1424

1425

1426

1427

1428

1429

1430

1431

1432

1433

1434

1435

- Accuracy: The RESPONSE is factually correct and provides relevant information. It correctly calculates the number of years ago Raymond's son was born based on the given information.
- Depth: The RESPONSE is sufficiently detailed and addresses the QUESTION fully. It provides a clear and concise answer to the problem.
- Clarity: The RESPONSE is clear, structured, and easy to understand. It uses simple language and avoids any ambiguous or confusing parts.
- Engagement: The RESPONSE is engaging and conversational, using a friendly and approachable tone. It provides a clear and concise answer to the problem, making it easy for the user to understand.
- Safety: The RESPONSE avoids offensive, harmful, toxic, or inappropriate content. It adheres to ethical and safety guidelines, promoting inclusivity and respect.

3. Suggestions for Improvement:

1436

1437

1438

1439

1440

1441

1442

1443

1444

- Provide a more detailed explanation of the calculation process, including the steps taken to arrive at the final answer. This would help users better understand the reasoning behind the solution.
- Consider providing alternative approaches or strategies for solving the problem, such as using a different method or providing a visual representation of the information.
- Ensure that the RESPONSE is consistent with the user's expectations and provides a clear and concise answer to the problem. This would help users better understand the information and make more informed decisions.

1445

1446

1447

1448

1449

In conclusion, the AI system has accurately understood the intent of the QUESTION and provided a high-quality RESPONSE. However, there are still opportunities for improvement, such as providing a more detailed explanation of the calculation process and considering alternative approaches or strategies for solving the problem. By addressing these areas, the AI system can deliver even better RESPONSES in the future.

1450

1451

1452

1453

1454

1455

1456

1457

Table 12: An Example of Model Self-Evaluation. The proposed solution contains an initial computational error (**Raymond was born 6 years before Samantha, he should be the older individual, implying his current age should be $31+6=37$ years**). Nevertheless, the model's Self-Evaluation process merely yields some uninformative assessments.

L QUALITATIVE EXAMPLES

Prompt:

1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511

What type of soil is suitable for cactus?

EVOLVE Direct Response:

Cacti, being succulent plants adapted to arid environments, thrive in well-draining soils that replicate their natural desert conditions. The ideal soil for cacti should meet these characteristics:

1. **Well-Draining:** Cacti cannot tolerate waterlogged conditions. The soil should have excellent drainage to prevent root rot.
2. **Low Nutrient Content:** Cacti prefer soils low in organic matter and rich in minerals, especially calcium and magnesium. Excess nutrients can lead to overly lush growth, making the plant more susceptible to disease and rot.
3. **Sand-Like Texture:** The soil should be composed of a mix of coarse sand, perlite, or pumice, which helps with drainage and aeration.
4. **Coarse-Grained:** Fine, compacted soils can retain water, which is detrimental to cacti. A gritty, well-aerated soil is essential.

Here's a typical cactus soil mix:

- **Coarse Sand:** About 40-50% (to ensure excellent drainage)
- **Pumice or Perlite:** About 30-40% (to improve aeration and drainage)
- **Compressed Coco Coir:** About 10-20% (to hold some moisture and provide a gentle nutrient base)
- **Calcitic Lime:** Optional (to raise the pH and provide calcium, essential for healthy growth)

You can purchase a pre-mixed cactus soil or create your own by combining these components. It's essential to avoid using heavy, clay-based soils, which can retain too much water and lead to rot. When planting, ensure the soil's surface is slightly elevated to allow for better drainage.

Always ensure the soil remains slightly dry between waterings to mimic the natural desert environment. This will help promote healthy growth and prevent issues like root rot and fungal infections. Happy cactus growing!

EVOLVE Self-Refinement Response:

The **ideal soil type** for cacti is well-draining, nutrient-poor, and slightly acidic to neutral (pH 6.5-7.5). Here's a detailed breakdown:

1. Composition:**• Primary Ingredients:**

- **Coarse grit or sand (40-50%):** Provides excellent drainage, preventing waterlogging which can lead to root rot.
- **Perlite or pumice (20-30%):** Improves aeration, retains some moisture, and enhances drainage further.
- **Loam or vermiculite (10-20%):** Adds a bit of fertility and retains some moisture, though still minimal.

• Optional Additives:

- **Calcined clay (1-5%):** Helps maintain soil structure and provide slight nutrient content.
- **Activated charcoal (1-2%):** Improves drainage, reduces odors, and filters out harmful substances.

2. Characteristics:

- **Well-draining:** Absolutely crucial. Cacti cannot tolerate waterlogged conditions.
- **Nutrient-poor:** Too much fertilizer can lead to excessive growth, making the cactus weak and susceptible to disease.
- **Acidic to slightly alkaline:** Most cacti prefer a neutral pH, but slightly acidic (around 6.5-7.0) is often ideal.
- **Loose and airy:** Allows roots to breathe and develop properly.

Instead of creating your own mix, it's highly recommended to use a commercially available cactus-specific potting mix.

These blends are formulated with the perfect proportions of the above ingredients and are readily available at garden centers and nurseries.

By using this well-draining, nutrient-poor soil, you'll create an optimal environment for your cacti to thrive, just like their natural desert home.

Table 13: A generated example on AlpacaEval 2 shows that both Direct Response Generation and Self-Refinement provide highly valuable responses for cactus planting.

1512

Prompt:

1513

Mick pays his teacher \$800 for 40 lessons worth 2 hours each. If this will be all he is going to pay for his lessons, how much did he receive?

1514

EVOLVE+RM Direct Response:

1515

Based on the information provided, Mick paid \$800 for 40 lessons, with each lesson lasting 2 hours. The calculation for the total time spent in lessons is as follows:

1516

1517

1518

1519

1520

$$\text{Total hours} = \text{Number of lessons} \times \text{Duration per lesson} = 40 \times 2 = 80 \text{ hours}$$

1521

1522

1523

1524

1525

Since Mick is only paying \$800 for these lessons and there's no additional payment mentioned, the amount he "received" (which seems like a bit of a misinterpretation in this context, as he's paying for the lessons, not receiving a sum) is the same as what he paid, which is \$800.

1526

1527

Therefore, Mick received (in the sense of what he paid) \$800.

1528

1529

1530

1531

If the question were framed differently, ensuring clarity on what "received" exactly means (like "the total value of the lessons he paid for"), then the answer would still be \$800. But considering the literal receipt of cash, it aligns with \$800 he paid.

1532

EVOLVE+RM Self-Refinement Response:

1533

Mick paid a total of **\$800** for his lessons. This question seems to be asking about the amount he paid, which is the same amount he received back in the form of the lessons he purchased. Therefore, Mick received **\$800**.

1534

1535

The question mixes concepts of payment and "receiving back" (which is usually more about value received), but since all the money was paid upfront, it logically means Mick received the full value of what he paid for (40 lessons at \$20 each = \$800). Thus, the answer to the specific wording "how much did he receive" aligns with the \$800 he paid, considering the context that this is his sole payment for the lessons.

1541

1542

To summarize, Mick received **\$800**.

1543

GPT4-1106-Preview Response:

1544

To determine the amount Mick received per lesson, we need to divide the total amount paid by the number of lessons he received.

1545

1546

Mick paid \$800 for 40 lessons.

1547

1548

So, $\$800 \div 40 \text{ lessons} = \20 per lesson .

1549

1550

Each lesson is 2 hours long, so to find out how much Mick received per hour, we divide the amount per lesson by the number of hours per lesson:

1551

1552

$\$20 \text{ per lesson} \div 2 \text{ hours per lesson} = \10 per hour .

1553

1554

Thus, Mick received \$10 per hour for his lessons.

1555

1556

Table 14: A case study on AlpacaEval 2 demonstrates that our approach, EVOLVE, allows the model to understand questions more deeply and comprehensively than the baseline GPT-4-1106-Preview. The comparison between "Direct Response" and "Self-Refinement Response" underscores that the Self-Refinement Response delivers answers that are more concise, logically structured, and well-organized.

1557

1558

1559

1560

1561

1562

1563

1564

1565

Prompt:

1566 if it takes 8 hours to fly to Singapore, stop over for an hour and a half and then fly 14 hours to London then what
 1567 is the total travel time if the plane is delayed in Singapore

EVOLVE+RM Direct Response:

1568 The total travel time, **including the stopover**, is **20 hours**.

1570 Here's the breakdown:

1572 1. **Direct flight to Singapore:** 8 hours
 1573 2. **Stopover:** 1.5 hours
 1574 3. **Direct flight to London:** 14 hours

1576 **Total:** 8 hours + 1.5 hours + 14 hours = **23.5 hours**

1577 **However**, you mentioned a **delay** in Singapore. If the delay is only during the stopover and doesn't
 1578 extend the overall travel time (e.g., the plane still reaches London within a reasonable timeframe), the total
 1579 **affected** travel time remains **23.5 hours**.

1580 If the delay causes significant changes to the flight schedule and extends the overall travel time, you'd need more
 1581 information about the specific delay duration to calculate the exact new total time.

1583 Therefore, if the delay is considered minimal and doesn't affect the London arrival time, the **total**
 1584 **travel time remains 23.5 hours**.

EVOLVE+RM Self-Refinement Response:

1586 Here's how to calculate the total travel time, considering a delay in Singapore:

1. Calculate the base travel time without delay:

- 1589 • Singapore to London: 8 hours (flight) + 1.5 hours (stopover) + 14 hours (second flight) = **23.5 hours**

2. Account for the delay in Singapore:

- 1591 • Let the delay be represented as 'x' hours.

3. Calculate the new total travel time:

- 1594 • New Singapore to London travel time: 8 hours (flight) + 1.5 hours (stopover) + x hours (delay) + 14
 1595 hours (second flight) = **23.5 + x hours**

1596 Therefore, the total travel time with the delay is **23.5 + x hours**.

1598 To get a specific number, you need to know the duration of the delay (x). For example, if the delay
 1599 is 2 hours:

- 1600 • Total travel time: **23.5 + 2 = 25.5 hours**

GPT4-1106-Preview Response:

1602 To calculate the total travel time, we need to add up all the segments of the journey along with any delays.

1603 Here's the breakdown:

1605 - Flight to Singapore: 8 hours
 1606 - Stopover in Singapore: 1.5 hours
 1607 - Flight to London: 14 hours

1608 Now, if the plane is delayed in Singapore, we have to add that delay to the stopover time. However,
 1609 you haven't specified the duration of the delay.

1611 Let's assume the delay is D hours. The total travel time would then be:

1612 Total travel time = 8 hours (first flight) + 1.5 hours (scheduled stopover) + D hours (delay in Singa-
 1613 pore) + 14 hours (second flight)

1615 If you provide the duration of the delay, I can give you the exact total travel time.

1617 Table 15: A case study on AlpacaEval 2 reveals that the Self-Refinement Response exhibits a more coherent
 1618 problem-solving logic, comparable to GPT-4-1106-Preview, whereas the Direct Response demonstrates more
 1619 fragmented and disorganized reasoning. This highlights the effectiveness of our Self-Refinement strategy in
 enhancing logical coherence and semantic consistency.

1620 M DOWNSTREAM TASK EVALUATION
16211622 To assess whether the iterative training process leads to models' catastrophic forgetting of general
1623 knowledge, we conducted complementary evaluations on various downstream tasks beyond our
1624 main experiments. We follow the established evaluation protocols in lm-evaluation-harness
1625 and present the results for all models in Tab. 16, where the number below each benchmark's name
1626 indicates the number of few-shot examples used during evaluations.1627 Our evaluation results on multiple downstream tasks, such as ARC-C and BoolQ, are consistent
1628 with observations reported in SimPO (Meng et al., 2024) and Self-Rewarding (Yuan et al., 2024).
1629 Specifically, our method does not lead to catastrophic forgetting during the post-training phase, as
1630 evidenced by the fact that the model's performance on these downstream tasks remains roughly
1631 similar to the baseline. Yuan et al. (2024) explain that this phenomenon may be due to the training
1632 data are based on Open Assistant prompts which may not be especially relevant to skills needed in
1633 downstream tasks, hence it is expected that the task performance stays roughly similar, or may even
1634 drop. Furthermore, this phenomenon is referred as an "**alignment tax**" by Ouyang et al. (2022).1635 N USE OF LLMs
16361638 We use LLMs only to refine the language and grammar in our paper. We do not use them for
1639 generating research ideas or for finding related work. We provide our complete original text to
1640 OpenAI's GPT-4o with instructions to make it more professional, coherent, and native-sounding for a
1641 research paper. We then carefully review all suggestions to guarantee that no factual content is altered
1642 and that all changes remain true to our original writing.1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673

1674
 1675
 1676
 1677
 1678
 1679
 1680
 1681
 1682
 1683
 1684
 1685
 1686
 1687
 1688
 1689
 1690
 1691
 1692
 1693
 1694
 1695
 1696
 1697
 1698
 1699
 1700
 1701
 1702
 1703
 1704
 1705
 1706
 1707
 1708
 1709
 1710
 1711
 1712
 1713
 1714
 1715
 1716
 1717
 1718
 1719
 1720
 1721
 1722
 1723
 1724
 1725
 1726
 1727

Table 16: Downstream task evaluation results on open leaderboard.

	ARC-C (25)	BoolQ (0)	CommonsenseQA (0)	HellaSwag (10)	LogiQA (0)	MMLU (5)	OpenBookQA (0)	PIQA (0)	WinoGrande (5)	Average
Llama-3.1-8B-Base	57.6	82.0	71.5	81.9	31.3	65.2	44.4	81.1	77.7	65.9
EVOLVE SFT	58.7	85.4	72.7	79.3	31.3	62.8	43.6	79.1	76.3	65.5
EVOLVE <i>iter1</i>	61.7	86.5	72.2	81.0	32.6	63.3	47.2	78.9	76.0	66.6
EVOLVE <i>iter2</i>	61.7	85.5	71.4	80.6	31.2	63.3	46.6	78.4	75.5	66.0
EVOLVE <i>offline</i>	62.2	86.4	72.5	80.9	33.0	63.6	46.0	79.6	76.5	66.7
EVOLVE_{rule} <i>iter1</i>	62.3	86.5	72.7	81.0	32.3	63.5	46.8	79.4	76.2	66.7
EVOLVE_{rule} <i>iter2</i>	61.6	86.4	72.3	81.1	31.8	63.6	46.4	79.3	75.9	66.5
EVOLVE_{rule} <i>iter3</i>	62.0	86.5	71.9	81.1	32.3	63.7	46.2	78.9	75.4	66.4
Iterative DPO SFT	59.4	86.9	73.4	78.6	32.9	60.0	44.0	79.6	77.1	65.8
Iterative DPO <i>iter1</i>	61.7	86.9	74.0	80.3	32.6	61.4	46.2	80.1	76.2	66.6
Iterative DPO <i>iter2</i>	61.4	86.8	74.0	80.2	31.2	61.6	45.8	79.3	76.0	66.3
Iterative DPO <i>offline</i>	61.4	87.0	74.0	80.3	32.3	61.2	46.0	80.1	76.4	66.5
Self-Rewarding <i>iter1</i>	61.3	86.9	74.0	80.2	32.4	61.2	45.8	80.3	76.2	66.5
Self-Rewarding <i>iter2</i>	61.4	87.0	74.4	80.3	32.7	61.2	46.2	80.3	76.0	66.6
Self-Rewarding <i>iter3</i>	61.2	87.0	74.0	80.3	32.7	61.3	46.0	80.4	76.6	66.6