# Probability-Consistent Preference Optimization for Enhanced LLM Reasoning

Yunqiao Yang[1,5*] Houxing Ren[1*] Zimu Lu[1] Ke Wang[1] Weikang Shi[1]
Aojun Zhou[1] Junting Pan[1,3] Mingjie Zhan[2†] Hongsheng Li[1,3,4 †]
[1]CUHK MMLab, [2]SenseTime Research
[3]CPII under InnoHK, [4]Shanghai AI Laboratory, [5]Zhiyuan College, SJTU
yangyunqiao7@gmail.com zhanmingjie@sensetime.com hsli@ee.cuhk.edu.hk

## Abstract

Recent advances in preference optimization have demonstrated significant potential for improving mathematical reasoning capabilities in large language models (LLMs). While current approaches leverage high-quality pairwise preference data through outcome-based criteria like answer correctness or consistency, they fundamentally neglect the internal logical coherence of responses. To overcome this, we propose Probability-Consistent Preference Optimization (PCPO), a novel framework that establishes dual quantitative metrics for preference selection: (1) surface-level answer correctness and (2) intrinsic token-level probability consistency across responses. Extensive experiments show that our PCPO consistently outperforms existing outcome-only criterion approaches across a diverse range of LLMs and benchmarks. Our code is publicly available at https://github.com/YunqiaoYang/PCPO.

## 1 Introduction

In recent years, enhancing the mathematical reasoning ability of Large Language Models (OpenAI, 2023; Anil et al., 2023; Touvron et al., 2023a,b; Bai et al., 2023; Jiang et al., 2023b, 2024; Anthropic, 2024; Yang et al., 2024a) (LLMs) has emerged as an important research direction (Ahn et al., 2024; Minaee et al., 2024). Among various approaches, Direct Optimization Preference (DPO) (Rafailov et al., 2024) is widely used due to its simplicity and efficiency. Since its introduction, numerous extensions of DPO have been proposed to further improve mathematical reasoning in diverse ways. For instance, methods such as Self-Rewarding LLMs (Yuan et al., 2024) and iterative DPO (Xu et al., 2023) demonstrate the effectiveness of iterative training strategies. Additionally, constructing high-quality pairwise preference data is essential for preference optimization (Bai et al., 2022; Yang et al., 2023).

To construct high-quality pairwise preference data, previous methods, such as IRPO (Pang et al., 2024) and ScPO (Prasad et al., 2024), select preference training pairs from generated responses that include a Chain-of-Thought (CoT) (Kojima et al., 2022) process followed by a final answer, have proven particularly effective in advancing mathematical reasoning performance. IRPO (Pang et al., 2024) employs gold labels (correct answers) to distinguish between chosen and rejected responses. Specifically, if a response's answer matches the gold label, it is designated as a chosen response; otherwise, it is classified as rejected. On the other hand, ScPO (Prasad et al., 2024) utilizes a voting function to evaluate the self-consistency (Wang et al., 2022) of responses. Responses whose answers appear most frequently are selected as chosen, while those with the least frequent answers are marked as rejected.

However, both methods focus solely on the correctness or frequency of the final answer while overlooking the internal logical connections or nuanced differences between responses. This limitation restricts the creation of more refined and informative preference training data (Wang et al., 2024). Consequently, models may have difficulty recognizing subtle yet critical distinctions between chosen and rejected responses during the iterative DPO training process (Fürnkranz and Hüllermeier, 2010; Wirth et al., 2017).

In this paper, we propose a novel method called **P**robability-**C**onsistent **P**reference **O**ptimization (PCPO), which leverages both the final answer and the internal logical connections of responses when selecting preference pairs. Our method is grounded in the principle that the token generation process in LLMs fundamentally involves predicting new tokens based on the highest conditional probability given all existing tokens (Vaswani, 2017;

[*] Equal contribution.
[†] Corresponding author.

*Findings of the Association for Computational Linguistics: ACL 2025*, pages 6435–6448
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

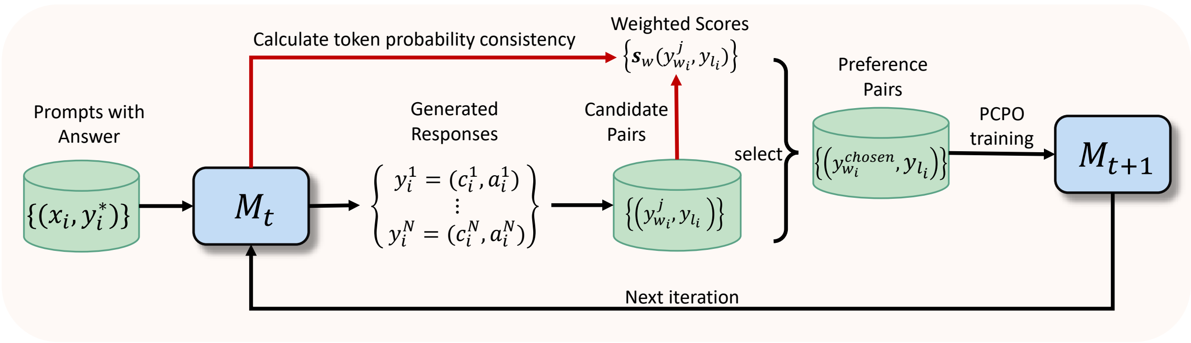


Figure 1: Overview of the PCPO method. The pipeline mainly consists of three steps. (1) Given a prompt set, utilize $M_t$ ($M_0$ as the seed model) to generate responses $y_i^n$ with reasoning $c_i^n$ and answer $a_i^n$, and construct candidate pairs on correctness §2.1. (2) utilize $M_t$ to calculate weighted scores $s_w$ for each pair based on the token probability consistency, and select preference pairs based on it §2.2. (3) train the next iteration model $M_{t+1}$ with the selected preference pairs and PCPO Loss §2.3.

Radford, 2018). Specifically, PCPO calculates a weighted score between preferred and dispreferred answers by evaluating the conditional probability of each token in the responses (Holtzman et al., 2019; Welleck et al., 2020). Preference pairs are then selected based on the highest weighted scores. This approach provides a quantitative framework for selecting preference pairs by considering not only the correctness of the final answer but also the internal coherence of the responses. By incorporating these factors, PCPO ensures a more robust and principled selection of preference pairs.

In each iteration, We first use the seed model to generate multiple responses for each math problem, and we construct a candidate preference pair set based on the correctness of the final answer. Then, we calculate the token-level consistency score for all the preference pairs from the candidate pair set. Afterward, we select the preference pairs with the highest token-level weighted score for each problem to construct preference training pairs eventually. Finally, we use the preference pairs selected to train the next iteration model with a modified DPO loss. To validate our method, we apply it to widely used math datasets, including GSM8K (Cobbe et al., 2021), MATH-500 (Hendrycks et al., 2021; Lightman et al., 2023) Olympiadbench (He et al., 2024) and AMC23 (Mathematical Association of America, 2023).To comprehensively demonstrate the effectiveness of our approach, we conduct experiments across a diverse range of seed models, such as Llama-3-8b-Instruct (Dubey et al., 2024), Mathstral-7b-v0.1 (Jiang et al., 2023a), Qwen-2.5-7B-Instruct (Yang et al., 2024b) and Qwen-2.5-Math-7B-Instruct (Yang et al., 2024c). Consistent results across these models showcase the effectiveness of our method.

In summary, our contributions are as follows:

1) We propose Probability-Consistent Preference Optimization (PCPO), a novel method that leverages both the final answers and the internal connections of the responses to select higher-quality preference pairs for training, thereby enhancing the mathematical reasoning capabilities of seed LLMs.

2) Extensive experiments demonstrate that our method consistently outperforms existing outcome-only criterion approaches (*e.g.*,, IRPO, ScPO) across a diverse range of LLMs and benchmarks.

3) Through empirical analysis, we highlight the critical importance of considering the internal connections of responses when selecting preference pairs. This insight paves the way for future research aimed at improving reasoning capabilities through more sophisticated preference pair selection methods.

## 2 Method

As depicted in Figure 1, our method starts with a pre-trained seed language model and a fixed prompt set of math problems with final answers. The PCPO pipeline mainly consists of three steps. (1) utilize $M_t$ ($M_0$ as the seed model) to generate responses $y_i^n$ with reasoning $c_i^n$ and answer $a_i^n$, and construct candidate pairs on correctness §2.1. (2) utilize $M_t$ to compute weighted scores $s_w$ for each pair based on the token probability, and select preference pairs based on it §2.2. (3) train the next iteration model $M_{t+1}$ with the selected preference pairs and PCPO loss §2.3. The model will be trained and updated at each iteration, resulting in a series of models $M_1, \ldots, M_T$.

### 2.1 Construct Candidate Pairs

We assume we have an initial model $M_0$, and a prompt set $D = \{(x_i, y_i^*)\}$ containing questions $x_i$ and their correct answers $y_i^*$. We focus on the process applied to a specific prompt $x_i$, and therefore, we omit the subscript $i$ for simplicity in the following subsections.

**Response Generation.** In each iteration, we first use the current model $M_t$ to generate N different responses for the prompt $x$, *i.e.,* $Y = \{y^n = (c^n, a^n) \sim M_t(x)\}$ and $n \in [N]$, where $c^n, a^n$ represents the Chain-of-Thought reasoning steps and the prediction answer. If the prediction answer $a^n$ of the response $y^n$ equals the gold answer $y^*$, we put the response into a response subset $Y_w$, otherwise $Y_l$.

**Prepare candidate pair set.** Assume that there are $p$ chosen responses $y_w$ in $Y_w$ and $q$ rejected responses $y_l$ in $Y_l$, where $p$ and $q$ satisfy the condition $p + q = N$. To generate all possible preference pairs, we use the cartesian product (Hewitt and Savage, 1955) between the set $Y_w$ and $Y_l$ to construct $p \times q$ pairs, denoted as $Y_w \times Y_l = \{(y_w, y_l) \mid y_w \in Y_w, y_l \in Y_l\}$. However, due to computational constraints, we must limit the number of candidate pairs. To achieve this, we employ the Levenshtein distance technique (Heeringa, 2004) to filter the candidate pairs effectively. Its rationale is discussed in Appendix A.

The Levenshtein distance measures the minimum number of edits required to transform one sequence into another, serving as a metric for sequence similarity. For each rejected response $y_l$, we compute its Levenshtein distance with all chosen responses $\{y_w\}$ and select the top $k$ pairs with the smallest distances as candidate pairs. This process results in the candidate pair set $\mathcal{C}_t^{pairs} = \{(y_w^j, y_l)\}$, where $j = 1, 2, ..., \min(p, k)$ represents the number of candidate chosen responses for each rejected response $y_l$.

### 2.2 Construct Preference Pairs

In this step, we first introduce the concept of *token probability consistency* ($c_t$), a token-level metric derived from the standard cross-entropy formulation for individual tokens (Vaswani, 2017; Radford, 2018; Hong et al., 2024):

$$\mathcal{L}_t = -\log P(x_i | x_{<i}). \tag{1}$$

Next, we define the *pair-weighted score*($s_w$), a pair-level metric computed from the *token probability consistency* values of the chosen and rejected responses within a pair. Based on this score, we selectively extract preference training pairs from the candidate pairs set $\mathcal{C}_t^{pairs}$.

**Calculate token probability consistency.** For each response $y$ in a certain candidate pair $(y_w, y_l)$, we perform the following steps. First we tokenize $y$ into a sequence $\{y^t\}$ using the tokenizer of the current iteration model $M_t$, where $t = 1, 2, ..., l$ and $l$ denotes the length of the token sequence $\{y^t\}$. We then infer $M_t$ to obtain the casual conditional probability $P_{M_t}(y^t | y^{<t}, x)$ for each token.

With the tokenized pairs and their corresponding token probabilities, we proceed to the next step: employing a matching function $\mathcal{M}$ in Appendix B to align the common tokens between the two responses in a pair sequentially. This allows us to compute the *token consistency score*($c_t$), which is defined as

$$\begin{aligned} c_t\left(y_w | y_l\right) = \exp(-| \log P_w\left(y_t | x, y_{<t}\right) \\ - \log P_l\left(y_t | x, y_{<t}\right) |)\ , y_t \in \mathcal{M}(y_w, y_l). \end{aligned} \tag{2}$$

The concept of comparing token-level losses draws inspiration from recent works such as Christopoulou et al. (2024), which emphasizes sparse token-level optimization, and Lin et al. (2024b), which highlights the importance of critical tokens in alignment tasks. Additionally, the use of exponential mapping aligns with the design principles of ORPO's odd-one-out loss (Hong et al., 2024), as both approaches aim to transform token-level differences into probabilistic metrics for more effective optimization. This combination of ideas provides a principled foundation for our token probability consistency framework.

**Calculating pair-weighted score.** The token-level consistency score $c_t$ is a normalized metric ranging between 0 and 1, where a higher value indicates a smaller difference in logarithmic probabilities between the chosen and rejected responses for a given token. Since the logarithmic probability represents a conditional probability, a higher score suggests that the preceding tokens provide the most relevant context for predicting the current token (Vaswani, 2017; Radford, 2018). To compute the overall score $s$, we aggregate the token-level consistency scores $c_t$ across all matched tokens. However, since the number of matched tokens varies with the length of the responses, we normalize the final score by dividing it by the total length of the preference pair. This yields the pair-weighted score $s_w$ for each preference pair, defined as

$$s_w\left(y_w|y_l\right)=\frac{\sum_t c_t\left(y_w|y_l\right)}{l_{y_l}}, \tag{3}$$

where $l_{y_l}$ denotes the length of the token sequence in the rejected response. This normalization ensures that the score is robust to variations in response length and provides a fair comparison across preference pairs.

**Select preference pairs.** Given that we have performed the process in Section 2.2 for all the candidate pairs $(y_w^j, y_l)$. Next, we select the preference pair with the highest pair-weighted score for the rejected responses $y_l$. Specifically, for a given rejected response $y_l$, we choose the corresponding chosen response $y_w^{chosen}$ that maximizes the pair-weighted score $s_w$. This ensures that the chosen response exhibits the strongest token-level consistency and correlation with the rejected response, making it the most suitable candidate for preference optimization (Holtzman et al., 2019; Welleck et al., 2020). The resulting set of selected preference pairs can be formally represented as

$$\mathcal{S}_t^{pairs}=\{(y_w^{chosen},y_l)\} \tag{4}$$

$$=\left\{\underset{(y_w^j y_l)}{\arg\max}\, s_w\left(y_w^j|y_l\right)\ \middle|\ y_w^j, y_l\in\{(y_w^j,y_l)\}\right\}, \tag{5}$$

where $\mathcal{S}_t^{pairs}$ denotes the final set of selected preference pairs for the prompt x, and $\arg\max$ identifies the chosen response $y_w^{chosen}$ that maximizes the pair-weighted score $s_w$ for a given rejected response $y_l$. This selection process ensures that the chosen pairs are optimized for token-level consistency and alignment with human preferences, while maintaining a strong correlation between the chosen and rejected responses.

### 2.3 PCPO Loss Function

We design our PCPO loss function as follows:

$$\mathcal{L}_{PCPO}(y^+,y^-|x)=$$

$$\underbrace{-s_w(x)\log\sigma\left(\beta\log\frac{M_\theta(y^+\,|\,x)}{M_t(y^+\mid x)}-\beta\log\frac{M_\theta(y^-\,|\,x)}{M_t(y^-\mid x)}\right)}_{\text{Weighted DPO Loss}}$$

$$\underbrace{-\frac{\alpha s_w(x)}{|y^+|}\log M_\theta(y^+\,|\,x)}_{\text{Weighted NLL Loss}}.$$

The loss function integrates a pair-weighted score $s_w$ into both DPO and NLL losses, inspired by IRPO (Pang et al., 2024) and ScPO (Prasad et al., 2024). The weighted DPO loss and the weighted NLL loss, dynamically prioritize pairs with high token-level consistency, akin to sparse alignment strategies in SparsePO (Christopoulou et al., 2024). It also adaptively balances language modeling with preference alignment, similar to ScPO's self-consistency weighting.

The use of $s_w$ as a dynamic weighting mechanism is grounded in token-level consistency principles from Zeng et al. (2024) and Lin et al. (2024b), while the inclusion of NLL loss ensures stable optimization, as highlighted in IRPO (Pang et al., 2024). This design enables adaptive sample weighting, robustness to sequence length variations, and flexible optimization through parameters $\beta$ and $\alpha$. The pair-weighted score $s_w$ serves as a key innovation, enhancing the training process's effectiveness and interpretability.

## 3 Experiment setup

**Datasets.** We assess the effectiveness of PCPO across a large and challenging range of mathematical reasoning datasets: **GSM8K** consists of 1.3k high-quality grade school math word problems. **MATH-500** is a curated subset drawn from the MATH dataset comprising 500 challenging competition-style mathematics problems. **Olympiadbench** is a test set of mathematics problems from olympiads, designed to assess deep problem-solving skills, creativity, and advanced mathematical reasoning. **AMC23** is a test set of 40 problems from the 2023 American Mathematics Competitions (AMC 12). These problems are renowned for their depth and subtlety, offering a rigorous assessment of reasoning skills and precision.

**Metrics.** We report zero-shot Pass@1 and Maj@8 results. The Pass@1 score denotes The greedy decoding accuracy of a single response. The Maj@8 score denotes the accuracy of the majority answer voted from 8 candidate responses (Wang et al., 2022). More evaluation details are presented in Appendix C.

**Training data.** Our training data includes 7.5k GSM8K training set, 7.5k MATH training set, 7.5k subset of Orca-math (Li et al., 2024), and 7.5k subset of Cn-k12 (Li et al., 2024), 30k in total. In our

| **Metric** | **GSM8K** | | **MATH-500** | | **Olympiadbench** | | **AMC23** | |
|---|---|---|---|---|---|---|---|---|
| Iteration | Pass@1 | Maj@8 | Pass@1 | Maj@8 | Pass@1 | Maj@8 | Pass@1 | Maj@8 |
| *Llama3-8B-Instruct* | | | | | | | | |
| Seed $M_0$ | 71.3 | 81.6 | 30.8 | 34.2 | 8.1 | 10.2 | 10.0 | 7.5 |
| IRPO $M_1$ | 79.1 | 86.4 | 29.4 | 35.6 | 7.3 | 10.0 | 0 | 17.5 |
| IRPO $M_2$ | 81.1 | 88.4 | 30.6 | 36.6 | 6.7 | 9.8 | 0 | 12.5 |
| ScPO $M_1$ | 79.3 | 87.5 | 30.2 | 34.6 | 6.4 | 10.4 | 7.5 | 15.0 |
| ScPO $M_2$ | 81.6 | 88.6 | 32.2 | 36.4 | 7.9 | 10.5 | 5.0 | 17.5 |
| PCPO (ours) $M_1$ | 80.1 | 87.8 | 32.2 | 36.6 | 7.9 | 9.5 | **15.0** | **22.5** |
| PCPO (ours) $M_2$ | **82.8** | **88.9** | **33.2** | **38.4** | **9.5** | **11.7** | 10.0 | 20.0 |
| *Mathstral-7B-v0.1* | | | | | | | | |
| Seed $M_0$ | 84.3 | 91.4 | 57.2 | 63.2 | 21.8 | 26.7 | 25.0 | 40.0 |
| IRPO $M_1$ | 87.0 | 92.3 | 57.2 | 63.4 | 23.6 | 29.0 | 20.0 | 32.5 |
| IRPO $M_2$ | 87.7 | 91.4 | 58.4 | 66.8 | 24.6 | 29.2 | 20.0 | 30.0 |
| ScPO $M_1$ | 87.1 | 92.0 | 57.4 | 65.4 | 23.4 | 30.5 | 22.5 | 27.5 |
| ScPO $M_2$ | 87.6 | 92.3 | 60.4 | 66.8 | 24.1 | 30.7 | 27.5 | 40.0 |
| PCPO (ours) $M_1$ | 87.9 | 91.9 | 58.6 | 66.4 | 24.9 | 29.2 | 20.0 | 37.5 |
| PCPO (ours) $M_2$ | **89.0** | **92.3** | **61.8** | **69.4** | **25.2** | **32.1** | **32.5** | **47.5** |
| *Qwen2.5-7B-Instruct* | | | | | | | | |
| Seed $M_0$ | 92.3 | 94.0 | 76.4 | 81.2 | 38.5 | 44.9 | 47.5 | 60.0 |
| IRPO $M_1$ | 92.2 | 93.9 | 75.2 | 80.4 | 37.9 | 43.3 | 50.0 | 55.0 |
| IRPO $M_2$ | 92.3 | 93.9 | 77.6 | 81.2 | 40.1 | 45.0 | 52.5 | 57.5 |
| ScPO $M_1$ | 92.2 | 94.1 | 76.8 | 80.8 | 39.9 | 44.4 | 55.0 | 60.0 |
| ScPO $M_2$ | 92.3 | 93.9 | 76.8 | 81.4 | 39.9 | 44.7 | **57.5** | 60.0 |
| PCPO (ours) $M_1$ | 92.6 | **94.5** | 76.4 | 81.8 | 39.9 | **45.9** | 45.0 | 62.5 |
| PCPO (ours) $M_2$ | **92.6** | 94.1 | **78.0** | **82.4** | **40.3** | 45.0 | **57.5** | **65.0** |
| *Qwen2.5-Math-7B-Instruct* | | | | | | | | |
| Seed $M_0$ | 92.9 | 93.9 | 81 | 83.0 | 43.4 | 46.1 | 62.5 | 70.0 |
| IRPO $M_1$ | 93.1 | 94.0 | 81.2 | 82.8 | 44.1 | 47.4 | 67.5 | 70.0 |
| IRPO $M_2$ | 92.7 | 93.9 | 79.8 | 83.6 | 44.6 | 47.7 | 65 | 70.0 |
| ScPO $M_1$ | 92.6 | 94.1 | 80.8 | 83.0 | 44.7 | 47.3 | 67.5 | 70.0 |
| ScPO $M_2$ | 93.1 | 94.0 | 80.8 | 83.0 | 44.6 | 48.1 | 67.5 | 70.0 |
| PCPO (ours) | 92.9 | **94.2** | 80.6 | 83.4 | **44.9** | 48.7 | **70.0** | 72.5 |
| PCPO (ours) | **93.3** | 94.1 | **81.4** | **83.8** | 44.3 | **48.7** | 67.5 | **75.0** |

Table 1: Results of our method PCPO comparing with the baseline methods on GSM8K, MATH, Olympiadbench, and AMC23. The results are zero-shot Pass@1 and Maj@8 accuracy.

approach, we don't need to generate new data, and the training data are fixed for all the experiments.

**Baselines.** **Seed Model** uses Chain-of-Thought prompting (Kojima et al., 2022) with greedy decoding, achieving zero-shot Pass@1 and Maj@8 accuracy. **IRPO** (Pang et al., 2024) utilizes iterative training with pairwise preferences at the outcome level, considering the correctness of the final answer when building preference training data. **ScPO** (Prasad et al., 2024) uses an inference-time-only approach that selects the most frequent final answer to build preference training data. Similar to IRPO, ScPO is still an outcome-level method that considers the correctness and the frequency of the final answer.

**Implementation details.** We set $N = 16$ to generate responses for the training data, with the temperature of 1 and top-$p = 0.95$. For each iteration, we sample 15k training data, training a total of 6 epochs with a useful batch size of 128. We use an initial leaning rate $1.0 \times 10^{-7}$ with the cosine scheduler and AdamW optimizer with a warm-ratio of 0.1 for smoother training. The NLL regularization coefficient $\alpha$ is set to 1 and the DPO loss term coefficient $\beta$ is set to 0.5, following Prasad et al. (2024). For the Pass @1 evaluation, we implement greedy decoding with the temperature of 0, and for the Maj@8 evaluation, we set a temperature of 0.95 and top-$p = 0.95$. We use one node containing 8 A800 GPUs for training.

## 4 Main Rresults

### 4.1 Comparison Results

The main results are shown in Table 1, demonstrating that the performance of our PCPO exceeds baseline methods across multiple seed models on the GSM8K, MATH, Olympiadbench, and AMC23 benchmarks.

Specifically, for the Llama-3-8B-Instruct model, PCPO achieves significant improvements over ScPO and IRPO. On the GSM8K Pass@1 test, it surpasses ScPO and IRPO by 1.2 and 1.7 points, respectively. Similarly, on the MATH-500 Pass@1 test, it outperforms these baselines by 1.0 and

| **Metric** | **GSM8K** | | **MATH-500** | | **Olympiadbench** | | **AMC23** | |
|---|---|---|---|---|---|---|---|---|
| Iteration | Pass@1 | Maj@8 | Pass@1 | Maj@8 | Pass@1 | Maj@8 | Pass@1 | Maj@8 |
| *Llama3-8B-Instruct* | | | | | | | | |
| Seed $M_0$ | 71.3 | 81.6 | 30.8 | 34.2 | 8.1 | 10.2 | 10.0 | 7.5 |
| IRPO+DPO $M_1$ | 79.8 | 88.1 | 29.4 | 34.2 | 6.8 | 10.8 | 5.0 | 10.0 |
| IRPO+DPO $M_2$ | 81.7 | 88.2 | 30.0 | 35.8 | 7.4 | 8.1 | 0 | 5.0 |
| ScPO+DPO $M_1$ | 79.3 | 86.7 | 29.4 | 37.0 | 7.3 | 11.0 | 5.0 | 10.0 |
| ScPO+DPO $M_2$ | 81.3 | 88.6 | 31.6 | 38.8 | 7.0 | 8.7 | 5.0 | 12.5 |
| PCPO (ours)+DPO $M_1$ | 80.6 | 87.9 | 30.4 | 38.4 | 7.4 | 11.0 | 7.5 | 10.0 |
| PCPO (ours)+DPO $M_2$ | **81.9** | **89.0** | **31.8** | **39.8** | **9.3** | **12.1** | **7.5** | **15.0** |
| PCPO (ours) $M_1$ | 80.1 | 87.8 | 32.2 | 36.6 | 7.9 | 9.5 | 15.0 | 22.5 |
| PCPO (ours) $M_2$ | 82.8 | 88.9 | 33.2 | 38.4 | 9.5 | 11.7 | 10.0 | 20.0 |

Table 2: DPO training results with the preference pair training data curated by our PCPO method and baseline methods on GSM8K, MATH, Olympiadbench, and AMC23. For instance, IRPO+DPO represents DPO training with the preference data constructed by IRPO method. The results are zero-shot Pass@1 and Maj@8 accuracy.

| **Metric** | **GSM8K** | | **MATH-500** | | **Olympiadbench** | | **AMC23** | |
|---|---|---|---|---|---|---|---|---|
| Iteration | Pass@1 | Maj@8 | Pass@1 | Maj@8 | Pass@1 | Maj@8 | Pass@1 | Maj@8 |
| *Llama3-8B-Instruct* | | | | | | | | |
| PCPO (ours)+DPO $M_1$ | 80.6 | 87.9 | 30.4 | 38.4 | 7.4 | 11.0 | 7.5 | 10.0 |
| PCPO (ours)+DPO $M_2$ | 81.9 | 89.0 | 31.8 | 39.8 | 9.3 | 12.1 | 7.5 | 15.0 |
| PCPO (ours) $M_1$ | 80.1 | 87.8 | 32.2 | 36.6 | 7.9 | 9.5 | **15.0** | 22.5 |
| PCPO (ours) $M_2$ | **82.8** | 88.9 | **33.2** | 38.4 | **9.5** | 11.7 | 10.0 | 20.0 |

Table 3: PCPO Loss training and original DPO Loss training results comparison. The results are zero-shot Pass@1 and Maj@8 accuracy.

2.6 points, respectively. The improvements are more pronounced on the OlympiadBench Pass@1 test, with gains of 1.6 and 2.8 points over ScPO and IRPO, respectively. Notably, on the AMC23 Pass@1 test, PCPO achieves an impressive lead of 7.5 and 15.0 points over ScPO and IRPO, respectively. A similar trend is observed for Mathstral-7B-v0.1, with PCPO achieving gains of 1.4 and 1.3 points on GSM8K Pass@1, 1.4 and 3.4 points on MATH-500 Pass@1, and 5.0 and 12.5 points on AMC23 Pass@1 over ScPO and IRPO.

For the Qwen-2.5-7B-Instruct model and Qwen-2.5-MATH-7B-Instruct model, the performance gains are relatively smaller. We provide a theoretical analysis based on some literature. McKenzie et al. (2023) propose that LMs may show inverse scaling or worse task performance with increased training data scale. And according to Gan and Liu (2024), the efficacy of large language models (LLMs) is extensively influenced by both the volume and quality of the training data. Qwen-2.5-7B-Instruct model and Qwen-2.5-MATH-7B-Instruct model utilized iterative fine-tuning of data and was reinforced by a reward model during the post-training phase (Yang et al., 2024b). As a result, the quality of the training dataset we use does not significantly benefit the LLM. Nevertheless, PCPO still consistently outperforms IRPO and ScPO across all benchmarks, demonstrating a clear advantage over outcome-level methods. For the Qwen-2.5-MATH-7B-Instruct model, while IRPO and ScPO underperform the seed model $M_0$ on MATH-500, PCPO continues to demonstrate consistent gains, highlighting its robustness over outcome-level methods.

Table 1 also demonstrated that the performance of PCPO shows more consistency and robustness over the iteration training, detailed explanations in Appendix D. Overall, PCPO consistently outperforms the baselines that rely solely on final results when constructing preference training data on all the benchmarks with Pass@1 and Maj@8 metrics.

## 5 Ablation Study and Analysis

### 5.1 Effect of Preference Data

To isolate the impact of training data quality, we design an experiment where all methods—PCPO, IRPO (Pang et al., 2024), and ScPO (Prasad et al., 2024)—use the same DPO loss function (Rafailov et al., 2024), despite their original loss functions differing as described in Section 2.3. This allows us to directly compare the effectiveness of the preference pairs generated by each method.

Table 2 shows the performance of Llama-3-8B-Instruct trained with preference pairs curated by PCPO, IRPO, and ScPO, all optimized using the

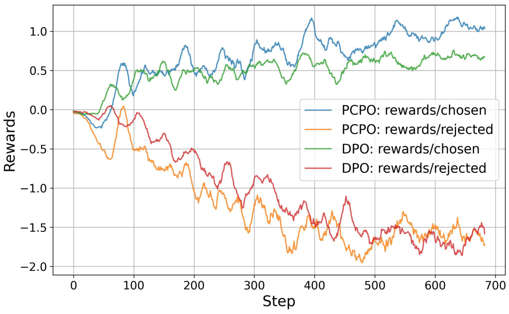


Figure 2: **Rewards of PCPO and DPO.** The chosen and rejected responses reward comparison of PCPO and DPO training on the same preference pairs.

DPO loss. Here, "IRPO+DPO" denotes training data curated by IRPO with the DPO loss, and similarly for other methods. The results demonstrate that models trained with PCPO's preference pairs consistently outperform those trained with IRPO or ScPO pairs. Specifically, PCPO $M_2$ achieves 1.9 and 2.3 points higher on the OlympiadBench Pass@1 test compared to IRPO $M_2$ and ScPO $M_2$, respectively, and 2.5 points higher on the AMC23 Pass@1 test than the best-performing model trained with IRPO or ScPO data. These results highlight the superior quality of PCPO's preference pairs, further validating its effectiveness in curating training data.

### 5.2 Effect of Loss

Table 3 demonstrates that the model trained with the PCPO Loss, as described in Section 2.3, outperforms the model trained with the original DPO Loss on the same PCPO curated preference pairs. Figure 2 shows the chosen and rejected responses reward comparison of PCPO and DPO training on the same preference pairs. The reward, denoted as $r = \beta \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}$, reflects the preference intensity of the current strategy model $\pi_\theta$ for generating a specific response **y** relative to the reference model $\pi_{ref}$ (Stiennon et al., 2020; Rafailov et al., 2024). Notably, the chosen reward for PCPO Loss exhibits a more pronounced increase, indicating more efficient learning from preference data due to an improved gradient update strategy. These results underscore that the PCPO Loss enables more effective preference training compared to the original DPO Loss.

### 5.3 Generalizability

Section 5.1 shows that the preference training data curated by our PCPO framework is of higher quality. To further validate its versatility, we apply our framework to enhance several DPO variants: RPO (Pang et al., 2024) (the single-iteration version of IRPO), IPO (Azar et al., 2024) (designed to prevent overfitting), ORPO (Hong et al., 2024) (reference-free alignment), and TDPO (Zeng et al., 2024) (token-level alignment). As shown in Table 4, PCPO +RPO, PCPO +IPO, PCPO +ORPO, and PCPO +TDPO consistently outperform their original counterparts across nearly all benchmarks. These results highlight the effectiveness and broad applicability of our framework in improving diverse preference alignment methods.

### 5.4 Case Study

We have already presented the efficiency and versatility of the preference training pairs curated with our PCPO framework in Section 5.1, 5.3, and we will quantitatively analyze it through some cases in this section. Figure 3 shows four responses from the Llama3-8B-Instruct with the same prompt generated in Section 2.1. These four responses exemplify the response generation process, showcasing both correct and incorrect answers, as well as various answer patterns. In this case, response-a and response-b have the right answer, while response-c and response-d have the wrong answer. Moreover, it can be easily seen from the **bold** sentence that these four responses have two answer patterns, and response-a and response-c are of one pattern while response-b and response-d are of another pattern. However, because they have no difference in their final answer, outcome-only methods are not able to distinguish them, so it's totally random for these methods to construct preference pairs from them. Our PCPO can easily identify different answer patterns in the token-level and put the responses with the nearest pattern in a preference pair.

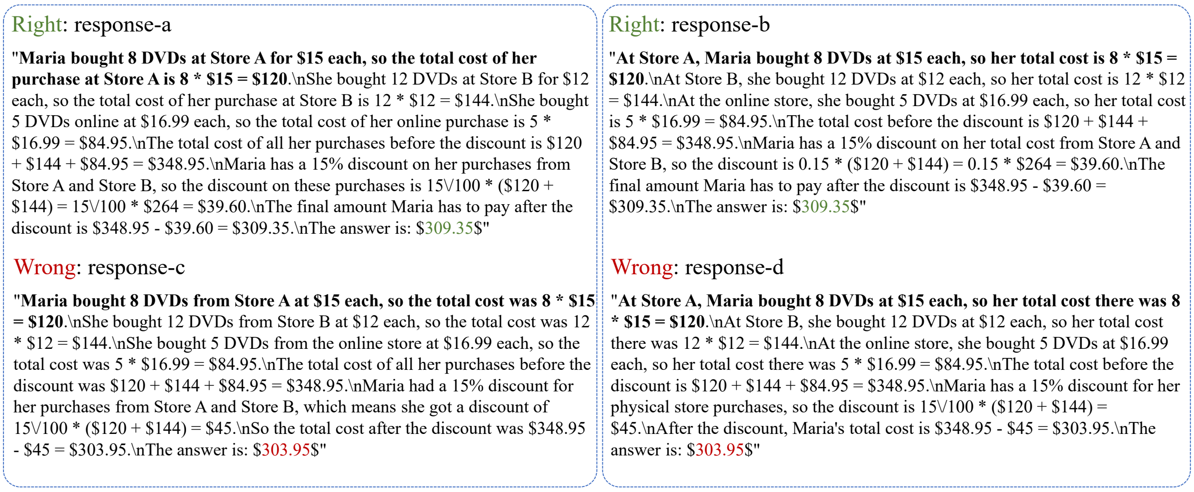
In this case, the weighted scores in Equation (3) of these pairs are shown in Table 5, thus PCPO is able to select response-a and response-c as a preference pair and response-b and response-d another. From this case study, we can conclude that our PCPO can select preference pairs with the highest token probability consistency, which the existing outcome-level methods can not do.

### 5.5 Analysis of training consumption

We conducted a statistical analysis of the quantitative comparisons with baseline methods. The entire training process can be divided into three parts: response generation, preference pair construction,

| **Metric** | **GSM8K** | | **MATH-500** | | **Olympiadbench** | | **AMC23** | |
|---|---|---|---|---|---|---|---|---|
| Iteration | Pass@1 | Maj@8 | Pass@1 | Maj@8 | Pass@1 | Maj@8 | Pass@1 | Maj@8 |
| *Llama3-8B-Instruct* | | | | | | | | |
| Seed $M_0$ | 71.3 | 81.6 | 30.8 | 34.2 | 8.1 | 10.2 | 10.0 | 7.5 |
| DPO $M_1$ | 79.8 | 87.4 | 29.4 | 34.2 | 6.8 | 10.8 | 5.0 | 10.0 |
| PCPO (ours)+DPO $M_1$ | **80.6** | **87.9** | **30.4** | **38.4** | **7.4** | **11.0** | **7.5** | **15.0** |
| RPO $M_1$ | 79.1 | 86.4 | 29.4 | 35.6 | 7.3 | 10.0 | 0 | 17.5 |
| PCPO (ours)+RPO $M_1$ | **80.0** | **87.3** | **30.6** | **37.4** | **7.4** | **10.7** | **5.0** | **22.5** |
| IPO $M_1$ | 80.6 | 88.0 | 24.4 | 37.6 | 8.1 | 11.9 | 10.0 | 10.0 |
| PCPO (ours)+IPO $M_1$ | **81.3** | **88.1** | **32.2** | **38.4** | **9.9** | **12.6** | **15.0** | **20.0** |
| ORPO $M_1$ | 81.6 | 88.1 | 27.0 | 32.8 | 8.0 | 10.5 | 10.0 | 12.5 |
| PCPO (ours)+ORPO $M_1$ | **81.9** | **88.2** | **29.0** | **36.4** | **8.6** | **11.9** | 10.0 | **25.0** |
| TDPO $M_1$ | **79.8** | 86.5 | 29.8 | 35.0 | 7.7 | 9.0 | 5.0 | 12.5 |
| PCPO (ours)+TDPO $M_1$ | 79.7 | **87.1** | **30.4** | **36.2** | **8.4** | **9.8** | 5.0 | **25.0** |

Table 4: Performance of applying PCPO framework to construct training data with different DPO variants on GSM8K, MATH, Olympiadbench, and AMC23. The results are zero-shot Pass@1 and Maj@8 accuracy.



Right: response-a

"**Maria bought 8 DVDs at Store A for $15 each, so the total cost of her purchase at Store A is 8 * $15 = $120**.\nShe bought 12 DVDs at Store B for $12 each, so the total cost of her purchase at Store B is 12 * $12 = $144.\nShe bought 5 DVDs online at $16.99 each, so the total cost of her online purchase is 5 * $16.99 = $84.95.\nThe total cost of all her purchases before the discount is $120 + $144 + $84.95 = $348.95.\nMaria has a 15% discount on her purchases from Store A and Store B, so the discount on these purchases is 15\/100 * ($120 + $144) = 15\/100 * $264 = $39.60.\nThe final amount Maria has to pay after the discount is $348.95 - $39.60 = $309.35.\nThe answer is: $309.35$"

Right: response-b

"**At Store A, Maria bought 8 DVDs at $15 each, so her total cost is 8 * $15 = $120**.\nAt Store B, she bought 12 DVDs at $12 each, so her total cost is 12 * $12 = $144.\nAt the online store, she bought 5 DVDs at $16.99 each, so her total cost is 5 * $16.99 = $84.95.\nThe total cost before the discount is $120 + $144 + $84.95 = $348.95.\nMaria has a 15% discount on her total cost from Store A and Store B, so the discount is 0.15 * ($120 + $144) = 0.15 * $264 = $39.60.\nThe final amount Maria has to pay after the discount is $348.95 - $39.60 = $309.35.\nThe answer is: $309.35$"

Wrong: response-c

"**Maria bought 8 DVDs from Store A at $15 each, so the total cost was 8 * $15 = $120**.\nShe bought 12 DVDs from Store B at $12 each, so the total cost was 12 * $12 = $144.\nShe bought 5 DVDs from the online store at $16.99 each, so the total cost was 5 * $16.99 = $84.95.\nThe total cost of all her purchases before the discount was $120 + $144 + $84.95 = $348.95.\nMaria had a 15% discount for her purchases from Store A and Store B, which means she got a discount of 15\/100 * ($120 + $144) = $45.\nSo the total cost after the discount was $348.95 - $45 = $303.95.\nThe answer is: $303.95$"

Wrong: response-d

"**At Store A, Maria bought 8 DVDs at $15 each, so her total cost there was 8 * $15 = $120**.\nAt Store B, she bought 12 DVDs at $12 each, so her total cost there was 12 * $12 = $144.\nAt the online store, she bought 5 DVDs at $16.99 each, so her total cost there was 5 * $16.99 = $84.95.\nThe total cost before the discount is $120 + $144 + $84.95 = $348.95.\nMaria has a 15% discount for her physical store purchases, so the discount is 15\/100 * ($120 + $144) = $45.\nAfter the discount, Maria's total cost is $348.95 - $45 = $303.95.\nThe answer is: $303.95$"

Figure 3: **A few right and wrong responses from the same prompt.** The four responses can be divided into two groups, where each has a similar response pattern.

| Pairs | a & c | b & c | a & d | b & d |
|---|---|---|---|---|
| $s_{\text{weighted}}$ | **0.791** | 0.525 | 0.559 | **0.793** |

Table 5: The weighted score $s_{weighted}$ of the four responses in Figure 3.

and training. The costs of the generating process and training process are nearly identical across all methods, with the main difference arising from the preference pair construction step. We calculated the average computational consumption (converted to 8*A800 GPU Hours) of training seed models for one iteration, as shown in table 6.

Although our method requires approximately 15% more GPU hours in the training process due to the need for token probability calculations, we believe it's worthy the marginal performance gain over the baseline methods.

## 6 Related Works

**Preference optimization for math reasoning.** Reinforcement learning from human feedback (RLHF) (Christiano et al., 2017) methods such as Direct Optimization Preference (DPO) (Rafailov et al., 2024) have emerged as a prominent approach for aligning Large Language Models (LLMs) with human preferences (Ouyang et al., 2022; Yang et al., 2024b). Recent advancements have introduced specialized variants for mathematical reasoning tasks. For instance, IRPO (Pang et al., 2024) selects preference training pairs from generated responses that include a Chain-of-Thought (CoT) and trains with DPO Loss adding a NLL term. ScPO (Prasad et al., 2024) utilizes a voting function to evaluate the self-consistency (Wang et al., 2022) of responses and trains with a weighted DPO+NLL loss. IPO aims to prevent DPO from overfitting to the preference dataset and ORPO eliminates the need for a reference model. our proposed PCPO distinguishes itself by explicitly considering the in-

| Process | Generate Responses | Construct Preference Pairs | Train | Total |
|---|---|---|---|---|
| IRPO | 3.2 | N/A | 4.5 | 7.7 |
| ScPO | 3.2 | N/A | 4.5 | 7.7 |
| PCPO | 3.2 | 1.2 | 4.5 | 8.9 |

Table 6: The average computational consumption among PCPO and baseline methods.

ternal logical relationships within preference pairs, offering a unique approach to preference optimization in mathematical reasoning tasks.

**Token-level preference optimization.** Recent advancements in token-level preference optimization have sought to address the inherent mismatch between sequence-level rewards and the token-level nature of LLM training and generation (Lin et al., 2024a). For instance, TDPO (Zeng et al., 2024) introduces a novel framework for aligning LLMs with human preferences at the token level, incorporating forward KL divergence constraints for individual tokens. SparsePO (Christopoulou et al., 2024) learns automatically during training inherently sparse masks over token-level rewards and KL divergences, highlighting that not all tokens are important in preference optimization. Lin et al. (2024b) illustrated the importance of critical tokens and proposed *c*DPO to automatically recognize and conduct token-level rewards for the critical tokens during the alignment process. The methods above either emphasize or ignore certain tokens when applying preference optimization, while our method PCPO utilizes token-level probability consistency to select preference pairs before the preference optimization process.

## 7 Conclusion

In this paper, we introduce Probability-Consistent Preference Optimization (PCPO), which provides a quantitative framework for selecting preference pairs by considering both the correctness of the final answer and the internal coherence of the responses. We introduced the concept of token probability consistency and the pair-weighted score to help select resulting preference training pairs. Extensive experiments demonstrate that our method consistently outperforms existing outcome-only criterion approaches (*e.g.,* IRPO, ScPO) across a diverse range of LLMs and benchmarks. This work paves the way for future research aimed at improving reasoning capabilities through more sophisticated preference pair selection methods.

## Limitations

While our approach demonstrates strong performance in supervised settings, it inherently depends on access to ground-truth final answers to construct reliable preference pairs. Acquiring high-quality labeled data is often resource-intensive, which restricts the scalability of our method to new domains. These limitations require preference optimization frameworks that can function effectively without gold-standard annotations. Addressing these challenges would significantly broaden the applicability of our method to real-world scenarios where labeled data is scarce or unavailable.

Additionally, the process of selecting preference training pairs necessitates generating a substantial number of candidate pairs, which in turn requires producing a larger volume of responses. This increases the computational demands and GPU hours, posing additional resource constraints.

## Ethics Statement

### Privacy Considerations

In this study, we employed several publicly available datasets, including GSM8K[1] (Cobbe et al., 2021), MATH-500[2] (Hendrycks et al., 2021; Lightman et al., 2023), Olympiadbench[3] (He et al., 2024), AMC23[4] (Mathematical Association of America, 2023), and Numina-math[5] (Li et al., 2024). These datasets are distributed under permissive licenses.

For model training, we utilized Llama-3-8B-Instruct (Dubey et al., 2024), Mathstral-7B-v0.1 (Jiang et al., 2023a), Qwen-2.5-7B-Instruct (Yang et al., 2024b), and Qwen-2.5-Math-7B-Instruct (Yang et al., 2024c). All these mod-

[1] https://huggingface.co/datasets/openai/gsm8k

[2] https://huggingface.co/datasets/HuggingFaceH4/MATH-500

[3] https://huggingface.co/datasets/realtreetune/olympiadbench

[4] https://github.com/QwenLM/Qwen2.5-Math/tree/main/evaluation/data/amc23

[5] https://huggingface.co/datasets/AI-MO/NuminaMath-CoT

els are licensed under Apache License 2.0 and are available for academic use.

In summary, our use of these datasets and models strictly complies with ethical guidelines for research data usage, upholding the principles of academic integrity and responsible research conduct.

### Security considerations

Security Considerations In this study, the models were trained using generated mathematical responses, which were carefully curated to ensure they do not contain any malicious or adversarial content. All responses were derived from fixed problem sets, which were explicitly selected to avoid any overlap with potential test datasets. This approach mitigates the risk of data leakage and ensures that the training process remains secure and unbiased. By adhering to these practices, we maintain the integrity of the training data and prevent any unintended exposure of sensitive or proprietary information.

## Acknowledgement

This study was supported in part by the Centre for Perceptual and Interactive Intelligence (CPII) Ltd., a CUHK-led InnoCentre under the InnoHK initiative of the Innovation and Technology Commission of the Hong Kong SAR Government, and in part by NSFC-RGC Project N_CUHK498/24. Hongsheng Li is a PI of CPII under the InnoHK.

## References

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*.

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernández Ábrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan A. Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vladimir Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, and et al. 2023. Palm 2 technical report. *CoRR*, abs/2305.10403.

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *CoRR*, abs/2309.16609.

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

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

Fenia Christopoulou, Ronald Cardenas, Gerasimos Lampouras, Haitham Bou-Ammar, and Jun Wang. 2024. Sparsepo: Controlling preference alignment of llms via sparse token masks. *arXiv preprint arXiv:2410.05102*.

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

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

Johannes Fürnkranz and Eyke Hüllermeier. 2010. Preference learning and ranking by pairwise comparison. In *Preference learning*, pages 65–82. Springer.

Zeyu Gan and Yong Liu. 2024. Towards a theoretical understanding of synthetic data in llm post-training: A reverse-bottleneck perspective. *arXiv preprint arXiv:2410.01720*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*.

Wilbert Jan Heeringa. 2004. Measuring dialect pronunciation differences using levenshtein distance.

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

Edwin Hewitt and Leonard J Savage. 1955. Symmetric measures on cartesian products. *Transactions of the American Mathematical Society*, 80(2):470–501.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

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

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

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023b. Mistral 7b. *CoRR*, abs/2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *CoRR*, abs/2401.04088.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. 2024. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13:9.

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

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, et al. 2024a. Rho-1: Not all tokens are what you need. *arXiv preprint arXiv:2404.07965*.

Zicheng Lin, Tian Liang, Jiahao Xu, Xing Wang, Ruilin Luo, Chufan Shi, Siheng Li, Yujiu Yang, and Zhaopeng Tu. 2024b. Critical tokens matter: Token-level contrastive estimation enhence llm's reasoning capability. *arXiv preprint arXiv:2411.19943*.

Mathematical Association of America. 2023. American Mathematics Competitions (AMC). `https://www.maa.org/math-competitions`.

Ian R McKenzie, Alexander Lyzhov, Michael Pieler, Alicia Parrish, Aaron Mueller, Ameya Prabhu, Euan McLean, Aaron Kirtland, Alexis Ross, Alisa Liu, et al. 2023. Inverse scaling: When bigger isn't better. *arXiv preprint arXiv:2306.09479*.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

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

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. *arXiv preprint arXiv:2404.19733*.

Archiki Prasad, Weizhe Yuan, Richard Yuanzhe Pang, Jing Xu, Maryam Fazel-Zarandi, Mohit Bansal, Sainbayar Sukhbaatar, Jason Weston, and Jane Yu. 2024. Self-consistency preference optimization. *arXiv preprint arXiv:2411.04109*.

Alec Radford. 2018. Improving language understanding by generative pre-training.

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

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Chenglong Wang, Yang Gan, Yifu Huo, Yongyu Mu, Qiaozhi He, Murun Yang, Tong Xiao, Chunliang Zhang, Tongran Liu, and Jingbo Zhu. 2024. Lrhp: Learning representations for human preferences via preference pairs. *arXiv preprint arXiv:2410.04503*.

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

Sean Welleck, Ilia Kulikov, Jaedeok Kim, Richard Yuanzhe Pang, and Kyunghyun Cho. 2020. Consistency of a recurrent language model with respect to incomplete decoding. *arXiv preprint arXiv:2002.02492*.

Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. 2017. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46.

Jing Xu, Andrew Lee, Sainbayar Sukhbaatar, and Jason Weston. 2023. Some things are more cringe than others: Preference optimization with the pairwise cringe loss. *arXiv preprint arXiv:2312.16682*.

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

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024b. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. 2024c. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. 2023. Rlcd: Reinforcement learning from contrast distillation for language model alignment. *arXiv preprint arXiv:2307.12950*.

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

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

# Appendix

## A Levenshtein Distance

In this experiment, we established an edit distance threshold of 8, corresponding to a total of 16 responses per prompt. For each rejected response, we retained the chosen responses with the 8 smallest edit distances. In cases where the number of chosen responses was fewer than 8, all available responses were preserved. Figure 4 displays the frequency distribution (bars) and cumulative percentage (line) of edit distance rankings (1–8, from min to max) for the final selected preference pairs. Rank 1 shows the highest frequency (50.2%), followed by rank 2 (20%), with frequencies declining sharply for ranks 5–8. The cumulative percentage reaches 95.4% by rank 5 and 100% at rank 8, indicating minimal contributions from higher ranks. Key Insights:

- Pareto Dominance: Ranks 1–5 (95.4% cumulative) dominate outcomes, aligning with the Pareto principle.
- Central Tendency: Rank 1 alone captures 50.2%, highlighting strong local consistency.
- Low Dispersion: Ranks 6–8 contribute negligibly (<4.6%), confirming high data concentration.

The results mean we can set an edit distance threshold of 5 to filter candidate pairs with more than 90 percent of resulting preference pairs within. This analysis supports algorithm optimization by prioritizing top-ranked edit distances for candidate pair filtering.

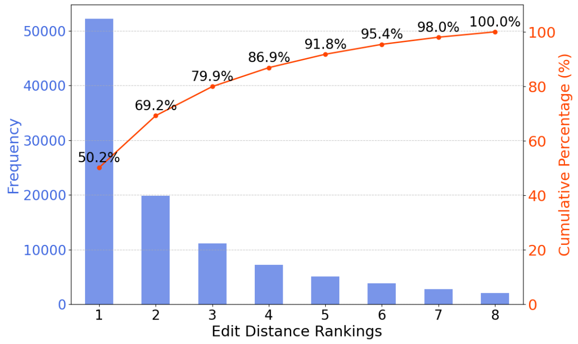


Figure 4: Frequency Distribution and Cumulative Percentage Pareto Chart of Edit Distance Rankings.

## B Matching Function

Algorithm 1 shows the match function pseudocode. Let $\mathbf{c} = [c_1, c_2, \ldots, c_m]$ and $\mathbf{r} = [r_1, r_2, \ldots, r_n]$ represent the token sequences of the chosen and rejected responses, respectively. The function identifies the longest common subsequence (LCS) of tokens between $\mathbf{c}$ and $\mathbf{r}$. For each aligned subsequence of length at least 1, it generates binary masks $\mathbf{M}_c \in \{0,1\}^m$ and $\mathbf{M}_r \in \{0,1\}^n$, Additionally, the function outputs an index mapping $\mathcal{I}$, which records the positions of aligned tokens in $\mathbf{c}$ and $\mathbf{r}$.

**Algorithm 1** Match Function

```
Require: $\mathbf{c} = [c_1, \ldots, c_m]$, $\mathbf{r} = [r_1, \ldots, r_n]$.
Ensure: Masks $\mathbf{M}_c$, $\mathbf{M}_r$, index mapping $\mathcal{I}$.
matcher ← SequenceMatcher(None, $\mathbf{c}$, $\mathbf{r}$)
$\mathbf{M}_c \leftarrow$ [False] $\times\, m$, $\mathbf{M}_r \leftarrow$ [False] $\times\, n$
$\mathcal{I} \leftarrow \emptyset$
for (tag, $i_1, i_2, j_1, j_2$) ∈ matcher do
    if tag = equal and $(i_2 - i_1) \geq 1$ then
        for (ci, rj) ∈ zip(range($i_1, i_2$), range($j_1, j_2$)) do
            $\mathbf{M}_c$[ci] ← True
            $\mathbf{M}_r$[rj] ← True
            $\mathcal{I} \leftarrow \mathcal{I} \cup \{(\text{ci}, \text{rj})\}$
        end for
    end if
end for
return $(\mathbf{M}_c, \mathbf{M}_r, \mathcal{I})$
```

In summary, the function can be compactly represented as

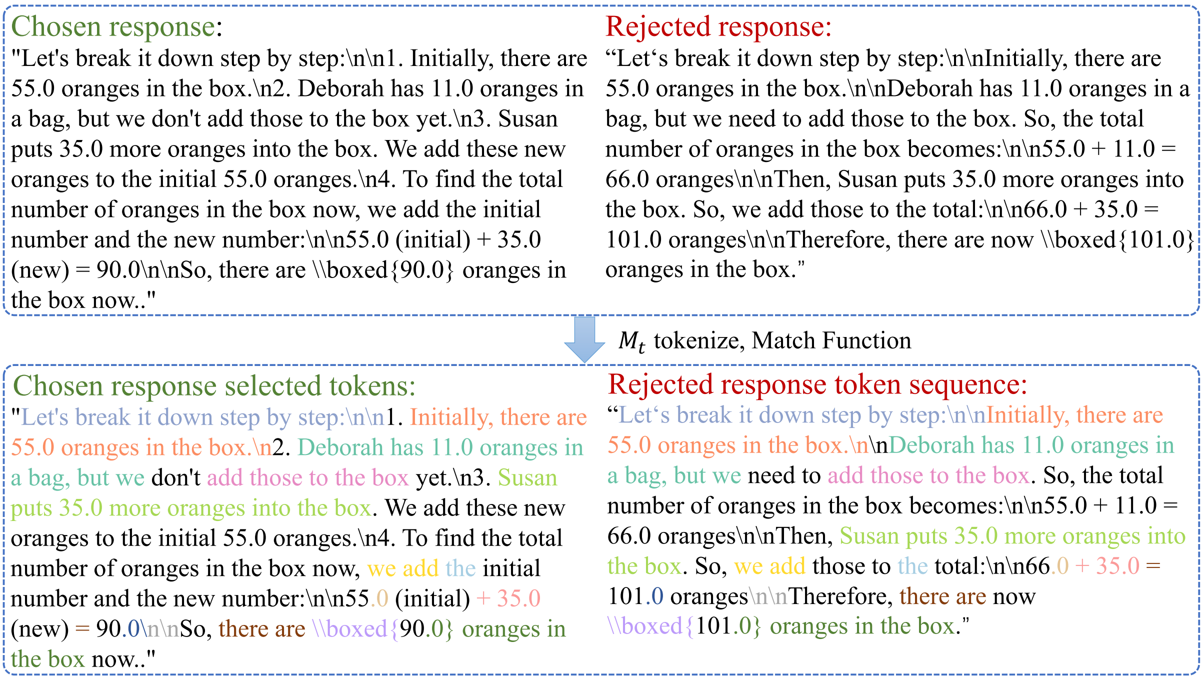
$$(\mathbf{M}_c, \mathbf{M}_r, \mathcal{I}) = \text{Match}(\mathbf{c}, \mathbf{r})$$

where Match is the sequence matching operation that identifies common tokens and generates the corresponding masks and index mapping.

Figure 5 illustrates the visualization for applying the Match function to align token sequences between chosen and rejected responses. As outlined in Section 2.2, we first tokenize the responses using the current iteration model $M_t$. Next, the Match function $\mathcal{M}$ generates common token masks for the sequences in a sequential manner. The masked tokens are highlighted in different colors, with index mappings indicating their positions in each sequence. We obtain the final matched tokens by extracting these tokens.

## C Evaluation details.

We use the standard automatic evaluation scripts following Qwen-Math (Yang et al., 2024c). The automatic evaluation pipeline mainly contains three steps: response generation, answer parsing, and comparison. First, the pipeline employs model $M_t$ to generate responses for each problem in the test set with a CoT prompt (Kojima et al., 2022)



Chosen response:

"Let's break it down step by step:\n\n1. Initially, there are 55.0 oranges in the box.\n2. Deborah has 11.0 oranges in a bag, but we don't add those to the box yet.\n3. Susan puts 35.0 more oranges into the box. We add these new oranges to the initial 55.0 oranges.\n4. To find the total number of oranges in the box now, we add the initial number and the new number:\n\n55.0 (initial) + 35.0 (new) = 90.0\n\nSo, there are \\boxed{90.0} oranges in the box now.."

Rejected response:

"Let's break it down step by step:\n\nInitially, there are 55.0 oranges in the box.\n\nDeborah has 11.0 oranges in a bag, but we need to add those to the box. So, the total number of oranges in the box becomes:\n\n55.0 + 11.0 = 66.0 oranges\n\nThen, Susan puts 35.0 more oranges into the box. So, we add those to the total:\n\n66.0 + 35.0 = 101.0 oranges\n\nTherefore, there are now \\boxed{101.0} oranges in the box."

Chosen response selected tokens:

"Let's break it down step by step:\n\n1. Initially, there are 55.0 oranges in the box.\n2. Deborah has 11.0 oranges in a bag, but we don't add those to the box yet.\n3. Susan puts 35.0 more oranges into the box. We add these new oranges to the initial 55.0 oranges.\n4. To find the total number of oranges in the box now, we add the initial number and the new number:\n\n55.0 (initial) + 35.0 (new) = 90.0\n\nSo, there are \\boxed{90.0} oranges in the box now.."

Rejected response token sequence:

"Let's break it down step by step:\n\nInitially, there are 55.0 oranges in the box.\n\nDeborah has 11.0 oranges in a bag, but we need to add those to the box. So, the total number of oranges in the box becomes:\n\n55.0 + 11.0 = 66.0 oranges\n\nThen, Susan puts 35.0 more oranges into the box. So, we add those to the total:\n\n66.0 + 35.0 = 101.0 oranges\n\nTherefore, there are now \\boxed{101.0} oranges in the box."

Figure 5: **The Match Function pipeline.** For a given pair of chosen and rejected responses, we first utilize the current iteration model $M_t$ to tokenize them and then use the algorithm 1 to get the longest common token subsequences, as highlighted in different colors.

(*e.g.,* "Please reason step by step, and put your final answer within boxed{ }") Second, the pipeline will extract the final answer from the response using regular expressions and fix the format of the answer such as removing extra brackets and modify the representation of fractions *etc.* Finally, the pipeline compares the extracted answer with the ground truth using an exact match criterion. This criterion requires that the answers satisfy one of the following conditions: (1) numerical equality, where both answers can be converted to floats and are equal, or (2) symbolic equality, where both answers can be converted to SYMPY[6] expressions and are equal. Through this pipeline, we can maximize the consistency and accuracy of the test results.

## D Iterations

Table 1 presents the model performance evolvement over the seed model $M_0$, $M_1$ and $M_2$. In a nutshell, PCPO performs better along the iterations over baseline methods IRPO, ScPO while achieving better absolute scores. For instance, on the Llama3-8B-Instruct,PCPO Pass@1 on the GSM8K test evolves from $M_1$ 80.1% to $M_2$ 82.8%, and the Olympiadbench test evolves from $M_1$ 7.9% to $M_2$ 9.5%, surpassing each iteration of the IRPO and ScPO method. Results on the Mathstral-7B-v0.1 show a similar trend. Although the iteration gains of all methods on the Qwen2.5-7B-Instruct and the Qwen2.5-MATH-7B-Instruct is less stable owning to the reason we explained in Section 4.1, our PCPO still performs a more consistent performance. The ScPO method almost saturates on the Qwen2.5-7B-Instruct through iterations, with only a small gain on the AMC23 Pass@1 test, and the IRPO method drops on the GSM8k, MATH-500 and AMC23 Pass@1 test from $M_1$ to $M_2$. In all, the performance of PCPO shows more consistency and robustness over the iteration training, confirming the effectiveness of our method.

## E Prompts

Prompt templates[7] for generating responses are shown below:

Response Generation Template

**User:**

```
Please reason step by step, and put
your final answer within \\boxed{{}}.

{{ question }}
```

**Assistant:**

[6] https://github.com/sympy/sympy

[7] The prompt template was from https://github.com/QwenLM/Qwen2.5-Math