Pruning Long Chain-of-Thought of Large Reasoning Models via Small-Scale Preference Optimization

Bin Hong[1], **Jiayu Liu**[1], **Zhenya Huang**[1]*, **Kai Zhang**[1], **Mengdi Zhang**[2]
[1] University of Science and Technology of China [2] Meituan
{hb2002, jy251198}@mail.ustc.edu.cn, {huangzhy, kkzhang08}@ustc.edu.cn

Abstract

Recent advances in Large Reasoning Models (LRMs) have demonstrated strong performance on complex tasks through long Chain-of-Thought (CoT) reasoning. However, their lengthy outputs increase computational costs and may lead to overthinking, raising challenges in balancing reasoning effectiveness and efficiency. Current methods for efficient reasoning often compromise reasoning quality or require extensive resources. This paper investigates efficient methods to reduce the generation length of LRMs. We analyze generation path distributions and filter generated trajectories through difficulty estimation. Subsequently, we analyze the convergence behaviors of the objectives of various preference optimization methods under a Bradley-Terry loss based framework. Based on the analysis, we propose Length Controlled Preference Optimization (LCPO) that directly balances the implicit reward related to NLL loss. LCPO can effectively learn length preference with limited data and training. Extensive experiments demonstrate that our approach significantly reduces the average output length by over 50% across multiple benchmarks while maintaining the reasoning performance. Our work highlights the potential for computationally efficient approaches in guiding LRMs toward efficient reasoning.

1 Introduction

Recently reasoning-oriented Large Language Models (LLMs), or Large Reasoning Models (LRMs) such as Deepseek-R1 [1] and QwQ-32B [2, 3] have achieved remarkable performance on complex tasks. These models learn to perform long Chain-of-Thought (CoT) thinking [4] through online reinforcement learning with verifiable reward (RLVR) (e.g., PPO [5] and GRPO [6] with conventional accuracy reward), which explicitly decomposes the given problem and proceeds to solve each subproblem in a step-by-step manner. Furthermore, they actively recall and verify previously generated steps throughout the reasoning process. This reasoning mode significantly increases the output length while empowering LLMs to effectively address reasoning-intensive tasks [7, 8].

The tendency of LRMs to generate excessively long responses introduces additional challenges [9–11]. As shown in Figure 1, for an easy question, DeepSeek-R1-Distill-Qwen-7B consumes 5465 tokens to solve it. On one hand, this tendency significantly increases computational and memory costs when using LRMs for various downstream tasks, which limits the applicability of continual learning for further enhancing the capabilities of LRMs. On the other hand, overly long outputs may lead to overthinking, where LRMs consume more tokens than standard models and may produce redundant or even incorrect outputs on relatively simple queries [12]. These challenges have prompted a renewed focus on a critical question for LRMs: *How to enable efficient reasoning with shorter output length and maintained reasoning performance?*

*Corresponding Author

Preprint. Under review.

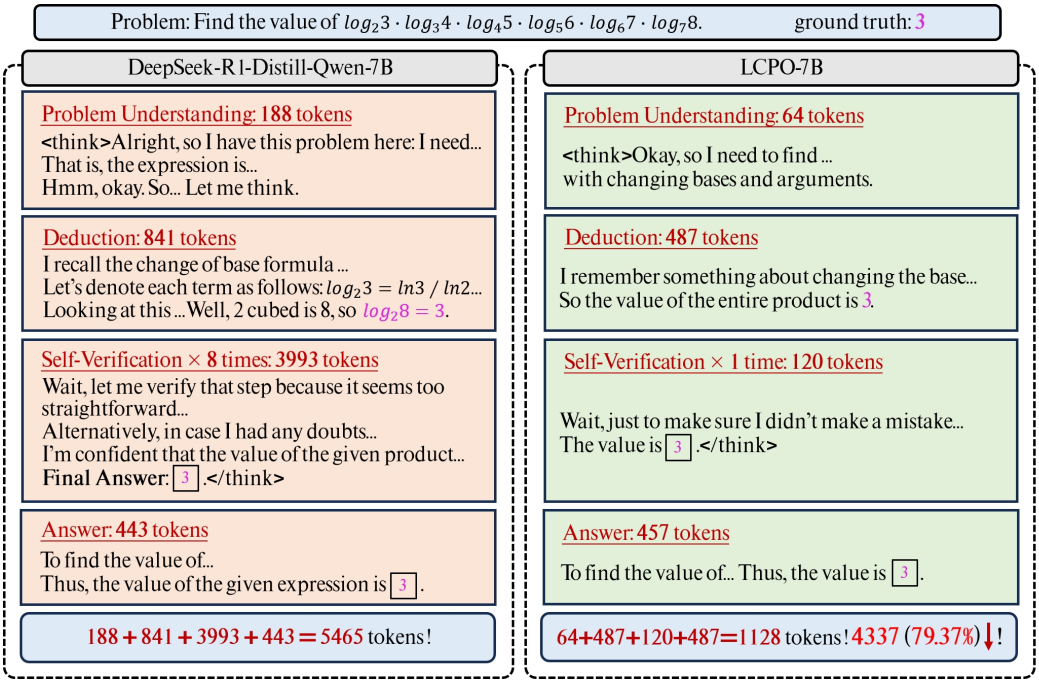


Figure 1: An example math problem from MATH-500. Left: the lengthy output of LRM. Right: the concise output of LRM after trained with LCPO.

Exploration of this question remains limited. Inference-time pruning methods [13, 14] append length-related tokens to the prompt or generation process to encourage earlier termination. While these approaches are low-cost, they are unstable in reducing length, and often compromise the model's reasoning capabilities. Subsequent works [15–18] adopt a large-scale multi-objective RL paradigm. They introduce a length-regularized reward alongside the conventional accuracy reward and perform RL on a large amount of data, aiming to reduce generation length and recover reasoning capacity. Online RL approaches involve complex training systems and higher resource demand compared to offline fine-tuning [19, 20], which contradicts the idea of efficient reasoning. Additionally, when incorporating budget-forcing mechanisms where a token limit is set in the prompt [14, 15], online RL-based methods are inherently constrained by manually predefined token budgets, limiting their adaptability to dynamic task requirements and can again compromise reasoning capabilities.

To address these limitations, we explore explicitly biasing the trajectory distribution within the LRMs generation space [21] in an offline manner. In this paper, we ask the following research questions:

- RQ1: Is there still shorter yet effective reasoning paths within the generation space of reasoning models?
- RQ2: How can we leverage limited training and data adjust reasoning model's generation distribution?

To answer these questions, we empirically analyze the trajectory distribution of generation space of LRMs. As illustrated in Figure 2a, we find that LRMs have inherently shorter yet effective reasoning trajectories, which can serve as a question-level self-adaptive training data source. We utilize the pass rate to estimate problem difficulty and leverage this metric to reduce data volume while reserving concise generation patterns. Subsequently, we employ preference optimization as an efficient self-distillation training method to push the output distribution toward the shortest effective paths. To find an ideal algorithm for effective small-scale training, we theoretically analyze the convergence conditions of different objective functions used in various preference optimization methods under a Bradley-Terry loss framework. As discussed in Appendix D, we find the implicit reward associated with NLL loss can hinder length preference equationment. Based on the analysis, we propose Length Controlled Preference Optimization (LCPO), where the implicit NLL-related reward is explicitly balanced by a counterpart term, enabling better equationment with response length preferences with

Table 1: Data needs for rollout and training of different efficient reasoning methods.

Method	Type	Data	Train Samples
CoD [13]	inference-time	-	-
L1 [15]	RL & budget-forcing	645k	645k
TrEff [18]	RL	24.8k	24.8k
DAST [22]	preference optimization	150k	20.6k
Ours	preference optimization	22k	0.8k

small-scale training data. Throughout this process, we establish an efficient reasoning pipeline that is both low-cost and computationally efficient.

We conducted extensive experiments using DeepSeek-R1-Distill-Qwen-1.5B and DeepSeek-R1-Distill-Qwen-7B, following previous works [15, 18], on six representative math reasoning benchmarks: MATH-500, GSM8K, Minerva-Math, AIME24, AMC23, OlympiadBench. Our method reduces the average generation length by over 50% across all benchmarks while maintaining the original model's reasoning performance. As illustrated in Figure 1, our approach can prune unnecessary parts of the reasoning trajectory and guides the model to more concise reasoning. Furthermore, as shown in 1, our method requires only 0.8k training samples and 50 training steps, largely reducing computational cost compared to prior approaches. Our findings and proposed method provide valuable insights for future research on efficient reasoning in large language models.

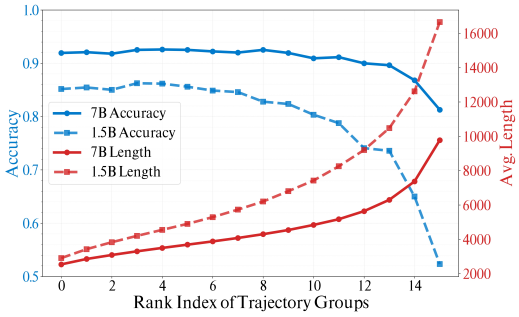
In summary, our main contributions are as follows:

- We analyze the generation space of reasoning models and propose a simple yet effective data filter strategy to reserve concise generation mode. We analyze the convergence behaviors of different preference optimization methods under a Bradley-Terry loss based framework.
- We propose Length Controlled Preference Optimization (LCPO) that is better for capturing preferences for response length from small-scale training data.
- We conducted extensive experiments on several reasoning benchmarks with models of varying parameter scales. The results demonstrate that our method reduces average output length by over 50% across different benchmarks while maintaining reasoning performance.

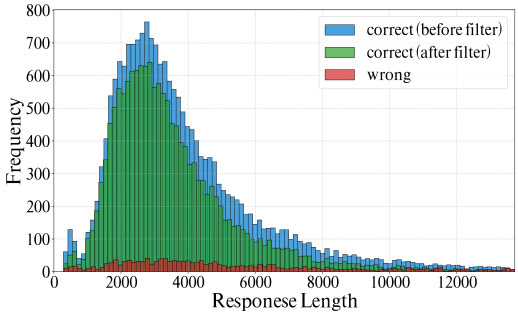
2 Related Work

Reinforcement Learning on LLM. Large Reasoning Models (LRMs) such as DeepSeek-R1 [1] and OpenAI' reasoning models [23, 24] employ reinforcement learning with verifiable reward (RLVR) to develop a model that exhibits strong reasoning capabilities. These models utilize online RL algorithms such as Proximal Policy Optimization [25, 5] (PPO) and Group Relative Policy Optimization [6] (GRPO), enabling environment interaction and reward acquisition. However, this approach requires interactive output generation and parameter updates, increasing training complexity. It typically demands more computational resources and is slower compared to off-policy fine-tuning methods. Direct Preference Optimization [26] (DPO) is a prominent off-policy alternative, introducing an implicit reward via pairwise preference data under the Bradley-Terry model. Subsequent advancements (e.g. SimPO [27], SimPER [28] and ORPO [29]) furthur eliminate the need for a reference model.

Test-Time Scaling and Efficient Reasoning. LRMs trained with RLVR exhibit strong reasoning capabilities [7, 8]. though they often generate excessively lengthy outputs—particularly in larger models like DeepSeek-R1 [9]. Recent studies [30–32] shows that Test-Time Scaling (increasing output length during inference) enhances problem-solving abilities, but at the cost of computational overhead and redundancy [11, 9]. To address this, explicit length penalties in online RL is naturally leveraged for shortening reasoning trajectories [16, 15]. This approach typically requires large-scale training with high resource demands but can introduce reasoning performance drop while incorporating with budget-forcing. Several studies [14, 33, 13, 18] mitigate excessive output length by incorporating length-related tokens or imposing explicit constraints on the maximum generation



(a) Performance of 7B and 1.5B models on LIMR with varying average output lengths.



(b) Distribution of output length from 7B models on LIMR before and after filtering

Figure 2: (a) As the number of output tokens increases, model performance tends to deteriorate. The x-axis represents the sorted rank of the samples. (b) After filtering, the distribution change is minor.

length, which often compromise models' performance. SFT-based methods such as Distill2-to-1 [34] and C3oT [35] construct CoT data of variable-length to shorten output length. DAST [22] designs a difficulty-based ranking function for preference data and leverage SimPO [27] for tuning.

3 Method

3.1 Overview

In this section, we present our pipline design step by step. We refine the training procedure from three key ingredients of RL: 1) Data. We harvest outputs from the generation space of the model, which contains only one seamless rollout process and uses a small-scale prompt data. 2) Reward, implicitly lies in the rank of data. We adopt an effective strategy to filter out data with high noise, and rank purely by length to maximize the preference signal. 3) Algorithm. We investigate the objectives of existing preference optimization methods, and introduce the Length Control Preference Optimization (LCPO) for more efficient length pruning.

3.2 Data and Reward: Empirical study on generation space

In this section, we aim to answer RQ1. We conduct experiments to analyze the trajectory distribution of generation space of LRMs. Throughout this process, we introduce our design on data and reward.

Experiment Setting and Notations. In this section, we conduct experiments on LIMR [36] with DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Qwen-1.5B [1] as reasoning models. For the two LRMs, questions in this dataset is relatively easy, and thus provides potentially shorter responses for training. For each question q_i in the dataset D, we generate 16 outputs and sort them by the number of tokens in the sequence. More details of generation can be found in Appendix B. This procedure samples a set of reasoning trajectories from the output space of model M:

$$\{(q_i, \{o_i^r\}_{r=1}^{16}, s_i)\}_{i=1}^{|D|}, \tag{1}$$

where o_i^r is the r-th shortest output of q_i, s_i denotes the average accuracy of question q_i over $\{o_i^j\}_{j=1}^{16}$.

First, we evaluate how the model's performance across the entire dataset varies with the r-th shortest reasoning trajectories $\{o_i^r\}_{i=1}^{D}$. As shown in Figure 2a, for outputs with longer average lengths, we observe a noticeable decline in the model's reasoning performance. For the 7B model, decline occurs for groups ranking higher than 10. For the 1.5B model, decline occurs earlier from ranking 7. In contrast, for outputs with shorter average lengths ranking 0 to 6, performance remains relatively stable despite variations in output length. This suggests that a reasoning model with a tendency for extended deliberation is still capable of performing shorter, more efficient reasoning without significantly compromising its accuracy. This highlights the potential of leveraging model-generated data to refine its behaviors.

Although model performance shows a clear decline with longer outputs, we argue that this is a correlation rather than a causal relationship. For reasoning-oriented models, their tendency to reflect and self-verify often leads to longer responses on incorrectly answered questions, as they typically perform more extensive trial-and-error reasoning [12, 37]. This suggests that output length is influenced by the model's perception of question difficulty, indicating that responses to more difficult questions may inherently contain more complex and more lengthy generation mode. To this end, we adopt a coarse and heuristic approach to approximate question difficulty and differentiate the conciseness of generated data. Specifically, we categorize each question q_i along with its outputs $o_{i\,r=1}^{r\,16}$ based on the corresponding score s_i:

$$\text{label}(q_i) = \begin{cases} \text{easy} & \text{if } s_i = 1, \\ \text{medium} & \text{if } 0 < s_i < 1, \\ \text{difficult} & \text{if } s_i = 0. \end{cases} \tag{2}$$

As shown in Figure 2b, correct responses account for the main part of the generation distribution. Thus, the filtering process does minor harm to the distribution of the spanned generation space. To guide the model to concise reasoning, we leverage the easy split of the training data. And we choose the shortest response as the chosen one, and the longest response as the rejected one to maximize the preference gap. Details about all the datasets can be found in Table 7.

3.3 Algorithm: Empirical Study of Preference Optimization Methods

In this section, we aim to answer RQ2. First, we provide brief preliminaries of preference optimization, which is further detailed in Appendix 3.3. Then we begin with a preliminary experiment to investigate the potential of existing preference optimization methods on effectively capturing length preferences. After that, we leverage Bradley-Terry model [38] to gain theoretical insight of the convergence characteristics of these methods. Finally, based on our findings and analysis, we introduce Length Controlled Preference Optimization (LCPO), an effective preference optimization method for shortening output length of reasoning models with small scale training.

Bradley-Terry Model. Let $\beta_i \in \mathbb{R}$ represent the ability value of a team i. Let the outcome of a game between teams i and j be determined by the difference of aibility values $\beta_i - \beta_j$. The Bradley-Terry model [38] defines the log-odds p_{ij} that team i beats team j as

$$\log \frac{p_{ij}}{1 - p_{ij}} = \beta_i - \beta_j, \tag{3}$$

Solving for p_{ij} yields

$$p_{ij} = \frac{1}{1 + e^{-(\alpha + \beta_i - \beta_j)}} = \sigma(\beta_i - \beta_j), \tag{4}$$

DPO and Preference Optimization Let x be the input sampled from the contexts space $\mathcal{X}$. We define a LLM θ as our policy $\pi_\theta(y|x)$, where the probability of generating a sequence y is $\pi_\theta(y|x)$. Reinforcement Learning from Human Feedback (RLHF) [5] uses unpaired data generated by the policy. It adds a KL divergence [39] penalty to the reward to restrict the policy distribution from changing too much. The objective is to maximize the following accumulated rewards:

$$L(\theta) = -\mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi_\theta(x,y)}\big[\, r(x,y) - \beta\, \mathbb{D}_{KL}(\pi_\theta(y|x) \parallel \pi_{\text{ref}}(y|x))\big]. \tag{5}$$

Direct Preference Optimization (DPO) [26] derives a implicit reward related to the policy:

$$r_{\text{DPO}}(x,y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} + \beta \log Z(x), \tag{6}$$

where $Z(x) = \sum_y \pi_{\text{ref}}(y|x) exp(\frac{1}{\beta} r(x,y))$ is the partition function that is hard to estimate. DPO leverage BT loss to remove $Z(x)$, resulting in the folllowing objective:

$$L_{\text{DPO}} = -\mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(x,y)}[\log \sigma(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)})]. \tag{7}$$

Following works design different rewards functions based on the policy model π_θ, resulting in different preference optimization objectives shown in Table 8.

Table 2: Results of different preference optimization methods. We report accuracy (Acc) and average number of tokens in the generation (Len). **Bold** indicates the best performance. $\Delta \downarrow \%$ denotes length reduction ratio (**higher is better**).

Method	MATH-500		GSM8K	
	Acc	Len ($\Delta \downarrow \%$)	Acc	Len ($\Delta \downarrow \%$)
Original [1]	92.20	4223	88.10	558
SFT [5]	87.00	3467 (17.90%)	84.99	**419 (24.91%)**
DPO [26]	90.40	2601 (38.41%)	88.02	464 (16.85%)
SimPO [27]	89.60	2885 (31.68%)	86.13	518 (7.17%)
SimPER [28]	91.20	3311 (21.60%)	86.88	576 (-3.23%)
ORPO [29]	90.00	2908 (31.14%)	85.44	467 (16.31%)
LCPO (Ours)	**92.00**	**1813 (57.07%)**	**89.76**	483 (13.44%)

3.3.1 Differences among preference optimizations

Experiment Setting. We train DeepSeek-R1-Distill-Qwen-7B on the easy split of LIMR and evaluate different preference optimization methods on MATH-500 [40] and GSM8K [41], which are widely adopted in recent studies [9]. The training configurations are provided in the Appendix B.

Table 2 summarizes our experiment result. Among existing methods on MATH-500, SimPO [27] and DPO [26] achieves superior performance with proper tuning on hyper-parameters, reducing average output length by 31.68% and 38.41% respectively. The conventional SFT approach [5] shows limited efficacy, yielding only a 17.90% length reduction. Notably, the recently proposed SimPER [28] underperforms with merely 21.60% reduction. ORPO [29] also demonstrates significant improvement (31.14%), achieving greater compression than SFT. On the GSM8K benchmark, while SFT and ORPO exhibit an accuracy degradation, they achieve more substantial length reductions (24.91% and 16.31% respectively).

3.3.2 Insight of convergence characteristics on preferences

Several methods demonstrate convergent behaviors derived from shared theoretical foundations. We further investigate where these convergence characteristics comes from and how they influence the alignment with preference.

We reformulate the objective functions of different methods into a log-sigmoid function form:

$$-\log \sigma(R(y_w, y_l, |x)) \tag{8}$$

The transformed objective can then be interpreted as a specialized BT model, where the sigmoid's parameters $R(y_w, y_l, |x)$ explicitly encode the reward difference between competing objective functions. During preference equationment, the model converges when the sigmoid output approaches 1, indicating near-deterministic preference satisfaction. Thus, we assess convergence by verifying the condition where $\sigma(R(y_w, y_l|x)) \to 1$. **The detailed derivations are provided in the Appendix D**.

Based on our analysis, we theoretically proved that the performance drop of ORPO is due to the NLL loss, which is the base of SFT. And we find that although SimPER can be helpful in non-reasoning tasks or relatively simple reasoning tasks (e.g. GSM8K [41]) as shown in [28], it is less effective in preference equationment of length control. DPO and SimPO have a more relaxed convergence condition, but highly rely on the hyper-parameters. Thus, they are unstable for limited training.

3.3.3 An objective better for convergence on preference data

Based on the analysis, we can conclude that two ingredients are needed in the context of length control: 1) Bradley-Terry loss with a relaxed reward margin and 2) well balanced negtive reward for the NLL loss. So first our objective follows the conventional BT loss form:

$$-\log \sigma(\beta r_\theta(y_w|x) - \beta r_\theta(y_l|x) + \epsilon). \tag{9}$$

We set ϵ to zero to let our model converge earlier for ingredient 1). Note that the reward of the NLL part can be shown explicit in the objective of a Bradley-Terry form:

$$L_{\text{NLL}} = -\frac{1}{|y|}\log \pi_\theta(y|x) = \log \sigma(\log \frac{p_\theta(y|x)}{1 - p_\theta(y|x)}) \tag{10}$$

Table 3: (Averaged) accuracy (Acc) and averaged number of tokens in generation (Len) of our method on different benchmarks. **Bold** denotes the best trade-off results. Δ denotes change on Acc. $\Delta\%$ denotes the change ratio of Len. *Total* denotes the total change on Acc. *Avg* denotes the average change ratio of Len.

Method	**MATH-500** Acc(Δ) Len($\Delta\%$)	**GSM8K** Acc(Δ) Len($\Delta\%$)	**Minerva-Math** Acc(Δ) Len($\Delta\%$)	**AIME24** Acc(Δ) Len($\Delta\%$)	**AMC23** Acc(Δ) Len($\Delta\%$)	**OlympiadBench** Acc(Δ) Len($\Delta\%$)	Total Avg
DeepSeek-R1-Distill-Qwen-1.5B							
Original	83.00 5665	81.88 1860	26.84 7211	29.17 16355	70.62 10162	44.66 11715	- -
CoD	81.80 (-1.20) 4788 (-15.48%)	80.89 (-0.99) 1488 (-20.00%)	25.74 (-1.10) 6231 (-13.59%)	26.67 (-2.50) 15465 (-5.44%)	69.84 (-0.78) 9052 (-10.92%)	43.03 (-1.63) 11014 (-5.98%)	-8.20 -11.90%
L1-Exact	81.80 (-1.20) 3335 (-41.13%)	82.26 (+0.38) 2992 (+60.86%)	25.37 (-1.47) 3673 (-49.06%)	21.25 (-7.92) 3665 (-77.59%)	67.19 (-3.43) 3406 (-66.48)	42.88 (-1.78) 3657 (-68.78%)	-15.42 -40.36%
L1-Max	82.20 (-0.80) 3374 (-40.44%)	83.40 (+1.52) 3134 (+68.49%)	27.57 (+0.73) 3418 (-52.60%)	22.29 (-6.88) 3523 (-78.46%)	**70.62 (0.00)** **3250 (-68.02%)**	**43.03 (-1.63)** **3452 (-70.53%)**	-7.06 -40.26%
TrEff	82.80 (-0.20) 2813 (-50.34%)	80.82 (-0.99) 809 (-56.51%)	25.74 (-1.10) 2950 (-59.10%)	28.54 (-0.63) 10719 (-34.46%)	69.84 (-0.78) 5166 (-49.16%)	43.62 (-1.04) 7652 (-34.68%)	-4.74 -47.38%
LCPO	**83.20 (+0.20)** **2397 (-57.69%)**	**81.58 (-0.30)** **787 (-57.69%)**	**27.21 (+0.37)** **2596 (-64.00%)**	**29.58 (+0.41)** **8810 (-46.13%)**	70.47 (-0.15) 4026 (-60.38%)	44.81 (+0.15) 4921 (-57.99%)	**+0.68** **-57.31%**
DeepSeek-R1-Distill-Qwen-7B							
Original	92.20 4223	88.10 558	36.76 5926	51.46 13411	87.97 6966	56.82 8789	- -
CoD	90.06 (-2.14) 2778 (-34.22%)	88.32 (+0.22) 416 (-25.45%)	36.40 (-0.36) 2733 (-53.88%)	48.54 (-2.92) 12029 (-10.30%)	87.34 (-0.63) 4880 (-29.95%)	54.60 (-2.22) 7200 (-18.08%)	-8.05 -28.65%
L1-Exact	89.80 (-2.40) 3555 (-15.82%)	90.90 (+2.80) 3345 (+499.46%)	36.03 (-0.73) 3341 (-43.62%)	21.04 (-30.42) 3442 (-74.33%)	69.53 (-18.44) 3337 (-52.10%)	53.86 (-2.96) 3712 (-57.77%)	-52.15 +42.64%
L1-Max	88.80 (-3.40) 2016 (-52.26%)	90.98 (+2.88) 1647 (+195.16%)	36.40 (-0.36) 1908 (-67.80%)	25.00 (-26.46) 3560 (-73.45%)	75.94 (-12.03) 3197 (-54.11%)	54.30 (-2.52) 2510 (-65.14%)	-41.89 -19.60%
TrEff	90.20 (-2.00) 2413 (-42.86%)	88.86 (+0.76) 484 (-13.26%)	37.13 (+0.37) 2917 (-50.78%)	48.13 (-3.33) 10204 (-23.91%)	87.03 (-0.94) 4420 (-36.55%)	53.56 (-3.26) 6970 (-20.70%)	-8.40 -31.34%
DAST	91.20 (-1.00) 3563 (-15.63%)	90.22 (+2.12) 1555 (+178.67%)	36.03 (-0.73) 8119 (+37.01%)	50.83 (-0.63) 15430 (+15.05%)	87.66 (-0.31) 6422 (-7.81%)	55.34 (-1.48) 10339 (+17.64%)	-2.03 +37.49%
LCPO	**92.00 (-0.20)** **1813 (-57.07%)**	**89.76 (+1.66)** **483 (-13.44%)**	**37.13 (+0.37)** **1797 (-69.68%)**	**51.04 (-0.42)** **6502 (-51.52%)**	**87.81 (-0.16)** **2812 (-59.63%)**	**55.79 (-1.03)** **4040 (-54.03%)**	**+0.22** **-50.90%**

where

$$p_\theta(y|x) = \exp(\frac{1}{|y|}\log \pi_\theta(y|x)),\ r_\theta(y|x) = log(\frac{p_\theta(y|x)}{1-p_\theta(y|x)}) \tag{11}$$

We directly balance this reward with a negative counterpart for ingredient 2). Then our proposed objective is as follow:

$$L_{\text{LCPO}} = -\lambda \log \sigma(\log(\frac{p_\theta(y_w|x)}{1-p_\theta(y_w|x)}) - \log(\frac{p_\theta(y_l|x)}{1-p_\theta(y_l|x)})), \tag{12}$$

where λ is a hyper-parameter.

4 Experiment

4.1 Main Result

Experiment Settings We choose DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Qwen-1.5B [1] as backbone reasoning models. We evaluate our method on the following math reasoning benchmarks: MATH-500 [40], GSM8K [41], Minerva-Math [42], AIME24 [43], AMC23 [44] and OlympiadBench [45]. We use accuracy as the metric for evaluation. Considering the limited size of AIME24 and AMC23, we sample 16 outputs for each question of them and compute their pass@1 score (e.g. accuracy averaged over 16 runs). Detailed information about all the benchmarks and baselines can be found in Appendix A. Detailed experiment settings can be found in Appendix B.

LCPO effectively reduce length while maintaining reasoning performance. As shown in Table 3, the experimental results demonstrate that our LCPO achieves consistent effectiveness across diverse benchmarks. With our method, the reasoning performance only drops slightly on a few benchmarks,

Table 4: Results of training on splits of different difficulty labels. $\Delta \downarrow \%$ denotes length reduction ratio (**higher is better**).

Method	MATH-500		GSM8K		Avg.
	Acc	Len ($\Delta \downarrow \%$)	Acc	Len ($\Delta \downarrow \%$)	chosen Len
Original Model	92.20	4223	88.10	558	-
LCPO w/ easy	92.00	**1813 (57.07%)**	89.76	483 (13.44%)	2232
w/ medium	91.80	2468 (41.56%)	87.11	**454 (18.64%)**	3637
w/ hard	92.00	3130 (25.88%)	88.02	496 (11.11%)	3681
w/o filter	92.40	3555 (15.82%)	88.63	588 (-5.38%)	3270

while the performance on most benchmarks are maintained. Compare to baseline methods, although LCPO falls short on reducing length on some benchmarks, it achieves a better overall trade-off between length and reasoning performance.

Besides, our approach significantly reduces generation length. We observe at least 46% reduction in generation length across all datasets except GSM8K, which is relatively easy. Overall, LCPO achieves more than 50% average length reduction over all benchmarks and yields the best overall trade-off, outperforming most baselines, while the computation demands are lower.

Notably, the 1.5B model consistently produces longer responses than the 7B model across nearly all datasets, while demonstrating greater length reduction with LCPO. This suggests the 1.5B model employs more complex reasoning patterns with higher variance in its trajectory distribution. Meanwhile, the limited effect of LCPO on the 7B model on GSM8K suggests that for tasks within the model's reasoning capability, the reasoning mode is less variable. LCPO can thus adaptively provide a smaller reduction on generation length to maintain the valuable information.

Distribution is shifted by preference optimization. Figure 3 shows the distribution of output length on MATH-500 before training and after training. It is clear that the peak of the distribution, which indicates the mean of the distribution is shifted left as expected. Meanwhile, after training, the variance is also decreased, resulting in a model with more consistency on length preference and equations better with the short effective trajectories in the generation space.

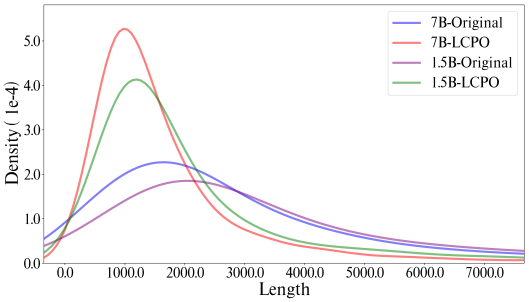


Figure 3: The distribution of output length on MATH-500 shifts after preference optimization.

4.2 Ablation Study

We conduct ablation experiments on the MATH-500 and GSM8K dataset to evaluate the key components of our designed pipeline. Following the framework outlined in 3.1, we investigate how the key ingredients (e.g. data with implicit reward and the preference optimization algorithm) affect the effectiveness of LCPO on length reduction.

Data with implicit reward. Table 4 focus on the data (with implicit reward), presenting the results of training on the split of different difficulty label, and the full dataset ("w/o filter"). As the length of the chosen responses in train set grows up, the average output length of model trained with LCPO also scales up, which equations with our assumption that responses to harder problems contains more complex reasoning mode, and confirms the effectiveness of our data filtering strategy. Although all the chosen responses of the hard split are not correct, the reasoning performance of the model is still improved. This indicates that LCPO focuses on capturing preference and is robust to the noise of correctness label of training data. Moreover, while the average length of chosen responses in the full dataset is not the longest, LCPO with the full dataset produce a model with the longest output length and a slightly better reasoning performance. It indicates that the diversity of generation mode can be important to reasoning through test-time scaling. Also, data distribution can be important for pruning long-CoT in LRMs.

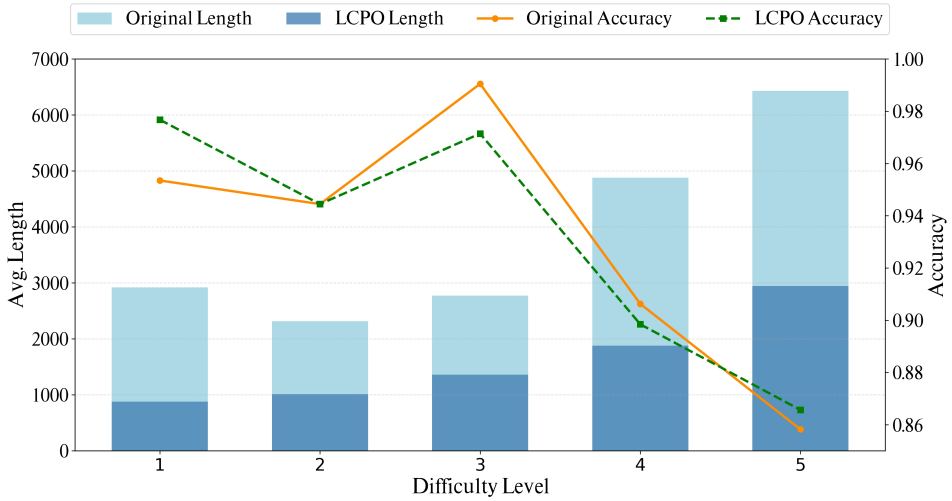


Figure 4: Changes of accuracy and generation length of DeepSeek-R1-Distill-Qwen-7B across different difficulty levels on MATH-500.

Algorithm of Preference Optimization. Table 2 shows the performance of LCPO comparing with various preference optimization methods. LCPO achieves the best performance of reducing length on MATH-500, while maintaining reasoning performance. Compared to SimPO, which is used in previous works [20, 22], LCPO performs better at the early training stage, while there is two less hyper-parameters to tune, which makes it easier to use. Notably, models trained with LCPO takes 50 steps and then stop, which uses 0.4k training data pairs, while models of other methods are trained with 350 steps.

4.3 Discussion

Generalizability to OOD scenario. To assess the effectiveness of LCPO to generalize in out-of-distribution scenarios, we conduct an evaluation on MMLU [46, 47], which is a massive multitask test consisting of 14K multiple-choice questions and covers 57 diverse domains, distinct from our training data in question form and subjects. Also, it likely falls outside our reasoning models' training distribution [15]. As shown in Table 5, models trained on math datasets with LCPO can still have 42.47% and 51.49% reduction of generation length, respectively.

Table 5: Results of LCPO on the out-of-distribution test set MMLU. ΔLen denotes the reduction ratio of average generation length.

Model	Method	Acc	Len	ΔLen
DS-1.5B	Original	34.67	1377	-
	LCPO	31.83	668	51.49%
DS-7B	Original	61.91	1321	-
	LCPO	61.61	760	42.47%

LCPO reduces generation length across different difficulty level. As shown in Figure 4, LCPO reduces over half of the generation length across different difficulty levels. At the same time, accuracy of different levels is preserved. Intuitively, easier problem need less tokens to process, which should result in shorter generation on level 1 problems. However, Figure 4 shows a longer average length on level 1 problems than leval 2 and level 3. This indicates a potential overthinking phenomenon of LRMs, where easier queries take more token cost and can result in a performance drop [11, 9, 20, 12]. While LCPO reduce the generation length on level 1, accuracy on level 1 improves. After training with LCPO, the average generation length is positively correlated with difficulty level, demonstrating consistency of reasoning effort with reasoning complexity.

5 Limitaions and Future Work

We discuss several limitations of our work in this section. 1) Following previous works [15, 22], our training set focuses on math tasks. The diversity of concise reasoning behaviors depends on the native

potential of LRMs. While experiments demonstrate the generalization ability to OOD scenarios, we believe there can achieve better results by carefully design a mixed training set of different reasoning modes. 1) Due to limited computational resources, we only conduct experiments on 1.5B and 7B models. Nevertheless, these extensive experiments demonstrate the effectiveness of LCPO across different model size.

6 Conclusion

In this paper, we investigated how to reduce the output length of large reasoning models via small-scale preference optimization. We proposed a pipeline for achieving trade-off between reasoning performance and output length using limited data. We distilled data from models' generation space, filtered data with question-level difficulty estimation. We analayzed different preference optimization methods under a unified Bradly-Terry loss framework and proposed Length Controlled Preference Optimization (LCPO). Extensive experiments demonstrated that our method reduced output length by a large margin while maintaining reasoning performance. Our work highlighted the potential of efficient methods for tuning model behaviors.

References

[1] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

[2] Q. Team, "Qwq-32b: Embracing the power of reinforcement learning," March 2025. [Online]. Available: https://qwenlm.github.io/blog/qwq-32b/

[3] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Tang, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, and Z. Qiu, "Qwen2.5 technical report," *arXiv preprint arXiv:2412.15115*, 2024.

[4] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[5] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Advances in neural information processing systems*, vol. 35, pp. 27 730–27 744, 2022.

[6] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu *et al.*, "Deepseekmath: Pushing the limits of mathematical reasoning in open language models," *arXiv preprint arXiv:2402.03300*, 2024.

[7] Q. Chen, L. Qin, J. Liu, D. Peng, J. Guan, P. Wang, M. Hu, Y. Zhou, T. Gao, and W. Che, "Towards reasoning era: A survey of long chain-of-thought for reasoning large language models," *arXiv preprint arXiv:2503.09567*, 2025.

[8] F. Xu, Q. Hao, Z. Zong, J. Wang, Y. Zhang, J. Wang, X. Lan, J. Gong, T. Ouyang, F. Meng *et al.*, "Towards large reasoning models: A survey of reinforced reasoning with large language models," *arXiv preprint arXiv:2501.09686*, 2025.

[9] S. Feng, G. Fang, X. Ma, and X. Wang, "Efficient reasoning models: A survey," *arXiv preprint arXiv:2504.10903*, 2025.

[10] X. Qu, Y. Li, Z. Su, W. Sun, J. Yan, D. Liu, G. Cui, D. Liu, S. Liang, J. He *et al.*, "A survey of efficient reasoning for large reasoning models: Language, multimodality, and beyond," *arXiv preprint arXiv:2503.21614*, 2025.

[11] Y. Liu, J. Wu, Y. He, H. Gao, H. Chen, B. Bi, J. Zhang, Z. Huang, and B. Hooi, "Efficient inference for large reasoning models: A survey," *arXiv preprint arXiv:2503.23077*, 2025.

[12] Y. Sui, Y.-N. Chuang, G. Wang, J. Zhang, T. Zhang, J. Yuan, H. Liu, A. Wen, S. Zhong, H. Chen *et al.*, "Stop overthinking: A survey on efficient reasoning for large language models," *arXiv preprint arXiv:2503.16419*, 2025.

[13] S. Xu, W. Xie, L. Zhao, and P. He, "Chain of draft: Thinking faster by writing less," *arXiv preprint arXiv:2502.18600*, 2025.

[14] N. Muennighoff, Z. Yang, W. Shi, X. L. Li, L. Fei-Fei, H. Hajishirzi, L. Zettlemoyer, P. Liang, E. Candes, and T. Hashimoto, "s1: Simple test-time scaling," in *Workshop on Reasoning and Planning for Large Language Models*, 2025.

[15] P. Aggarwal and S. Welleck, "L1: Controlling how long a reasoning model thinks with reinforcement learning," *arXiv preprint arXiv:2503.04697*, 2025.

[16] K. Team, A. Du, B. Gao, B. Xing, C. Jiang, C. Chen, C. Li, C. Xiao, C. Du, C. Liao *et al.*, "Kimi k1. 5: Scaling reinforcement learning with llms," *arXiv preprint arXiv:2501.12599*, 2025.

[17] B. Hou, Y. Zhang, J. Ji, Y. Liu, K. Qian, J. Andreas, and S. Chang, "Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning," *arXiv preprint arXiv:2504.01296*, 2025.

[18] D. Arora and A. Zanette, "Training language models to reason efficiently," 2025. [Online]. Available: https://arxiv.org/abs/2502.04463

[19] Q. Yu, Z. Zhang, R. Zhu, Y. Yuan, X. Zuo, Y. Yue, T. Fan, G. Liu, L. Liu, X. Liu *et al.*, "Dapo: An open-source llm reinforcement learning system at scale," *arXiv preprint arXiv:2503.14476*, 2025.

[20] X. Chen, J. Xu, T. Liang, Z. He, J. Pang, D. Yu, L. Song, Q. Liu, M. Zhou, Z. Zhang *et al.*, "Do not think that much for 2+ 3=? on the overthinking of o1-like llms," *arXiv preprint arXiv:2412.21187*, 2024.

[21] Y. Yue, Z. Chen, R. Lu, A. Zhao, Z. Wang, S. Song, and G. Huang, "Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model?" *arXiv preprint arXiv:2504.13837*, 2025.

[22] Y. Shen, J. Zhang, J. Huang, S. Shi, W. Zhang, J. Yan, N. Wang, K. Wang, and S. Lian, "Dast: Difficulty-adaptive slow-thinking for large reasoning models," *arXiv preprint arXiv:2503.04472*, 2025.

[23] A. Jaech, A. Kalai, A. Lerer, A. Richardson, A. El-Kishky, A. Low, A. Helyar, A. Madry, A. Beutel, A. Carney *et al.*, "Openai o1 system card," *arXiv preprint arXiv:2412.16720*, 2024.

[24] OpenAI, "Introducing openai o3 and o4-mini," https://openai.com/index/introducing-o3-and-o4-mini/, 2025, accessed: August 13, 2025.

[25] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[26] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," *Advances in Neural Information Processing Systems*, vol. 36, pp. 53 728–53 741, 2023.

[27] Y. Meng, M. Xia, and D. Chen, "Simpo: Simple preference optimization with a reference-free reward," *Advances in Neural Information Processing Systems*, vol. 37, pp. 124 198–124 235, 2024.

[28] T. Xiao, Y. Yuan, Z. Chen, M. Li, S. Liang, Z. Ren, and V. G. Honavar, "Simper: A minimalist approach to preference alignment without hyperparameters," in *The Thirteenth International Conference on Learning Representations*, 2025.

[29] J. Hong, N. Lee, and J. Thorne, "Orpo: Monolithic preference optimization without reference model," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024, pp. 11 170–11 189.

[30] J. Wei, X. Wang, D. Schuurmans, M. Bosma, b. ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 24 824–24 837. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf

[31] X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. H. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," in *The Eleventh International Conference on Learning Representations*, 2022.

[32] Q. Zhang, F. Lyu, Z. Sun, L. Wang, W. Zhang, Z. Guo, Y. Wang, I. King, X. Liu, and C. Ma, "What, how, where, and how well? a survey on test-time scaling in large language models," *arXiv preprint arXiv:2503.24235*, 2025.

[33] Z. Zeng, Q. Cheng, Z. Yin, Y. Zhou, and X. Qiu, "Revisiting the test-time scaling of o1-like models: Do they truly possess test-time scaling capabilities?" *arXiv preprint arXiv:2502.12215*, 2025.

[34] P. Yu, J. Xu, J. E. Weston, and I. Kulikov, "Distilling system 2 into system 1," in *The First Workshop on System-2 Reasoning at Scale, NeurIPS'24*, 2024.

[35] Y. Kang, X. Sun, L. Chen, and W. Zou, "C3ot: Generating shorter chain-of-thought without compromising effectiveness," in *Proceedings of the AAAI Conference on Artificial Intelligence*, no. 23, 2025, pp. 24 312–24 320.

[36] X. Li, H. Zou, and P. Liu, "Limr: Less is more for rl scaling," *arXiv preprint arXiv:2502.11886*, 2025.

[37] Z. Liu, C. Chen, W. Li, P. Qi, T. Pang, C. Du, W. S. Lee, and M. Lin, "Understanding r1-zero-like training: A critical perspective," 2025. [Online]. Available: https://arxiv.org/abs/2503.20783

[38] R. A. Bradley and M. E. Terry, "Rank analysis of incomplete block designs: I. the method of paired comparisons," *Biometrika*, vol. 39, no. 3/4, pp. 324–345, 1952.

[39] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[40] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe, "Let's verify step by step," in *The Twelfth International Conference on Learning Representations*, 2023.

[41] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman, "Training verifiers to solve math word problems," *arXiv preprint arXiv:2110.14168*, 2021.

[42] A. Lewkowycz, A. Andreassen, D. Dohan, E. Dyer, H. Michalewski, V. Ramasesh, A. Slone, C. Anil, I. Schlag, T. Gutman-Solo *et al.*, "Solving quantitative reasoning problems with language models," *Advances in neural information processing systems*, vol. 35, pp. 3843–3857, 2022.

[43] A. Online, "Art of problem solving," https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions, 2025, accessed: August 13, 2025.

[44] M. A. of America, "American mathematics competitions," https://maa.org/student-programs/amc/, 2025, accessed: August 13, 2025.

[45] C. He, R. Luo, Y. Bai, S. Hu, Z. Thai, J. Shen, J. Hu, X. Han, Y. Huang, Y. Zhang *et al.*, "Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 3828–3850.

[46] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[47] D. Hendrycks, C. Burns, S. Basart, A. Critch, J. Li, D. Song, and J. Steinhardt, "Aligning ai with shared human values," *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[48] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt, "Measuring mathematical problem solving with the math dataset," *NeurIPS*, 2021.

[49] M. Luo, S. Tan, J. Wong, X. Shi, W. Y. Tang, M. Roongta, C. Cai, J. Luo, L. E. Li, R. A. Popa, and I. Stoica, "Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl," https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2, 2025, notion Blog.

[50] G. Cui, L. Yuan, Z. Wang, H. Wang, W. Li, B. He, Y. Fan, T. Yu, Q. Xu, W. Chen *et al.*, "Process reinforcement through implicit rewards," *arXiv preprint arXiv:2502.01456*, 2025.

[51] D. P. Kingma, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[52] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[53] Y. Zheng, R. Zhang, J. Zhang, Y. Ye, Z. Luo, Z. Feng, and Y. Ma, "Llamafactory: Unified efficient fine-tuning of 100+ language models," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*. Bangkok, Thailand: Association for Computational Linguistics, 2024. [Online]. Available: http://arxiv.org/abs/2403.13372

[54] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica, "Efficient memory management for large language model serving with pagedattention," in *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

[55] F. Jelinek, R. L. Mercer, L. R. Bahl, and J. K. Baker, "Perplexity—a measure of the difficulty of speech recognition tasks," *The Journal of the Acoustical Society of America*, vol. 62, no. S1, pp. S63–S63, 1977.

A Brief Introduction of Datasets and Baselines

Math datasets We use 6 math datasets covering in domain and out of domain data for evaluation. The detailed information is as follows:

- MATH-500 [40] is a high-quality math problem solving dataset containing 500 problems extractd from the MATH [48] test set. It is widely used for evaluation of LLM reasoning [1, 3, 2].
- GSM8K [41] is a famous dataset containing 1319 primary school level math problems. It is widely used for LLM evaluation for years [5, 24].
- Minerva-Math [42] is a specialized collection designed for training and evaluating AI models on challenging mathematical reasoning tasks. It includes 272 problems ranging from algebra and calculus to advanced proofs. It is adopted by [49] for evaluation on LRMs trained from online RL.
- AIME24 [43] is a collection of challenging mathematical problems from the American Invitational Mathematics Examination (AIME) [43] held in 2024. It contains 30 extremely challenging problems from real world and is widely used for evaluating powerful LRMs. It is becoming the most popular dataset for olympiad-level math problem sovling evaluation [49].
- AMC23 [44] is a collection of challenging mathematical problems designed for the American Mathematics Competitions (AMC). It is also a widely used dataset.
- OlympiadBench [45] is an Olympiad-level bilingual multimodal scientific benchmark, featuring 8,476 problems from Olympiad-level mathematics and physics competitions, including the Chinese college entrance exam. We use the math split in English for test following previous works [49, 15].

General datasets We adopt the widely used dataset MMLU [46] for test. MMLU is a massive multitask test consisting of 14k multiple-choice questions from various branches of knowledge, spanning subjects in the humanities, social sciences, hard sciences, and other areas that are important for some people to learn. This covers 57 tasks including elementary mathematics, US history, computer science, law, and more. To attain high accuracy on this test, models must possess extensive world knowledge and problem solving ability.

Training datasets We use LIMR [36] dataset for training. LIMR is a subset of the MATH [48] training set containing 1389 math problems extracted by assessing consistency of accuracy reward scores during online RL. In our LCPO practice, we perform rollout on the whole set, while using only 400 problems from it for training 50 steps. Notably, switching to other datasets (e.g. the AIME-AMC split from the PRIME dataset [50]) does not significantly compromise the effectiveness of LCPO.

Baselines We evaluate baseline methods listed in 1. These baselines span across various types, covering inference-time method, RL method and budget forcing methods. If there is a special generation setting described in the original paper of a baseline (e.g. evaluation prompt), we adopt the original setting. Otherwise we use the settings described in B.

Table 6: Hyperparameters for various preference optimization methods.

Method	learning rate	preference weight	others
SFT	1e-5	-	-
DPO	5e-6	1	-
SimPO	5e-7	2	$\gamma = 0.5$
SIMPER	5e-7	-	-
ORPO	5e-6	0.2	-
LCPO (Ours)	5e-6	0.3	-

All the baselines are evaluated using the open-source model released by authors. Detailed information about the baselines are as follows:

- CoD [13], Chain-of-Draft prompting, is a training-free inference-time method performed by add an special instruction asking the model to keep a draft of at most 5 words for each thinking step.
- L1 [15] is a online RL based method incorporating with budget-forcing. The model is trained on a massive amount of data from the DeepScaleR dataset [49]. L1 provide two versions of model. L1-Exact is trained under a accurate length constrained reward penalty. L1-Max is trained under a ceiling limit of length constrain. We use the open-source weight from their official HuggingFace repository[2].
- TrEff [18] is another powerful online RL based method. During training, TrEff adopt a normalized length reward similar with and advantage estimation of DeepSeek's GRPO [6]. We use the open-source model released in the official repository[3].
- DAST [22] is a preference optimization based method. It use pass rate to estimate difficulty of the questions and then calculate a token length budget. A budget-based length penalty is applied when ranking preference data to achieve adaptive slow thinking. The original paper use DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Qwen-32B [1] as the base models. Their official repository[4] only contains the 7B model, which we adopt for evalutaion.

B Implementaion Details

Common settings All the experiments are done on 4× A800-80G PCIe. If there is no special notation, we train the model with LCPO for 3 epoch and save checkpoints for every 50 steps, using the checkpoint at the 50th step for every methods, which uses only 0.4k sample pairs. Other methods in Table 2 use the checkpoint at the 350th step, which involves 2.8k training samples. For all the experiment, we set batch size per GPU to 2, resulting in a equivalent batch size of 8. Context length is limited to 2.3k. Learning rate scheduler type is always cosine. Warmup ratio is always 0.1. Optimizer is always AdamW [51] from torch [52]. Dtype is always bfloat16.

For rollout and evaluation. If there is no special notation, we set temperature to 0.6. And the max tokens limit is always 32768 in all the experiments following [1]. For datasets with limited size (e.g. AIME24, AMC23), we report averaged accurcy over 16 samples (e.g. pass@1). Otherwise we report accurcy results. Besides, we report the average number of tokens as the average generation length which is denoted as *Len*.

Hyperparameters for various preference optimization metheds For every method in 3.3, we tuning the hyperparameters across a wide range, and present the best results we get. The specific settings for each method is shown in Table 6.

Infrastructures We use the wonderful llamafactory [53] for training. And we use vllm [54] for rollout. The evaluation code is based on [49]. Besides, we integrate evaluation code from repos of different datasets for their seperate evaluation (e.g. OlympiadBench [45]).

[2]https://huggingface.co/collections/l3lab/l1-67cacf4e39c176ca4e9890f4

[3]https://huggingface.co/daman1209arora/models

[4]https://github.com/AnonymousUser0520/AnonymousRepo01

Table 7: Statistics of all the datasets generated from LIMR.

Model	split	question num.	avg. chosen length	avg. rejected length
DeepSeek-R1-Distill-Qwen-7B	easy	988	2232	6662
DeepSeek-R1-Distill-Qwen-7B	medium	330	3637	15820
DeepSeek-R1-Distill-Qwen-7B	hard	52	3681	13107
DeepSeek-R1-Distill-Qwen-7B	easy-best	988	3089	6662
DeepSeek-R1-Distill-Qwen-7B	easy-pl	349	3245	16643
DeepSeek-R1-Distill-Qwen-7B	all	1389	3270	7015
DeepSeek-R1-Distill-Qwen-1.5B	easy	488	2078	8371

Prompts We adopt the prompt setting from [1], which is recommanded in the official repo. We keep this setting across all the experiments. Evaluation prompt is shown in Figure 5.

Evaluation Prompt

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

Figure 5: Evaluation prompt.

Dataset generate from LIMR In this paper, we leverage a heuristic way of performing self-distillation (rollout), filtering and ranking to data. Throughout this process, we get three split: easy, medium and hard, with respect to different models used for rollout. Here we list all the statistics of the dataset generate along this line in Table 7.

C Introducion for Preference Optimization

In this section, we first outline the foundational concepts and notations used throughout our study of preference optimization, including the formulation of pairwise preference data, reward modeling, and the learning objectives employed in various algorithms such as DPO, SimPO, and others. Then we analyze the convergence behavior of objectives of different preference optimization methods in a unified form based on Bradley-Terry loss [38] (BT loss).

Bradley-Terry Model. Let $\beta_i \in \mathbb{R}$ represent the ability value of a team i. Let the outcome of a game between teams i and j be determined by the difference of aibility values $\beta_i - \beta_j$. The Bradley-Terry model [38] take the log-odds corresponding to p_{ij} that team i beats team j as

$$\log \frac{p_{ij}}{1 - p_{ij}} = \alpha + \beta_i - \beta_j, \tag{13}$$

where α is an intercept term. Solving for p_{ij} yields

$$p_{ij} = \frac{1}{1 + e^{-(\alpha+\beta_i-\beta_j)}} = \sigma(\alpha + \beta_i - \beta_j), \tag{14}$$

where $\sigma(\cdot)$ is the sigmoid function. Under the LLMs setting, given an input x within a finite space of contexts $\mathcal{X}$, we define a policy as $\pi_\theta(y|x)$. For a pair of actions (y_i, y_j), we obtain a partial preference relation $y_i \succeq y_j$ between them through human annotations or similar methods, indicating y_i is prefered by ground truth compared to y_j. Tuning the policy for preference on a dataset $\mathcal{D} = \{(x^{(i)}, y_w^{(i)}, y_l^{(i)})\}_{i=1}^{|\mathcal{D}|}$ aims to maximize the the expected log-probability of $y_w^{(i)} \succeq y_l^{(i)}$. We take the rewards $r(x^{(i)}, y_w^{(i)})$ and $r(x^{(i)}, y_w^{(i)})$ of $y_w^{(i)}$ and $y_l^{(i)}$ given by a rule or reward model as their ability values. Then applying BT model gives

$$L(r, \mathcal{D}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}[\log \sigma(r(x, y_w) - r(x, y_l)]. \tag{15}$$

Table 8: Objective Functions of Preference Optimization Methods (Expectation Omitted)

Name	Objective
SFT [5]	$-\frac{1}{\lvert y_w\rvert}\log \pi_\theta(y_w\lvert x)$
DPO [26]	$-\log\sigma\left(\beta\log\frac{\pi_\theta(y_w\lvert x)}{\pi_{\text{ref}}(y_w\lvert x)}-\beta\log\frac{\pi_\theta(y_l\lvert x)}{\pi_{\text{ref}}(y_l\lvert x)}\right)$
SimPO [27]	$-\log\sigma\left(\frac{\beta}{\lvert y_w\rvert}\log\pi_\theta(y_w\lvert x)-\frac{\beta}{\lvert y_l\rvert}\log\pi_\theta(y_l\lvert x)-\gamma\right)$
SimPER [28]	$-\exp\left(\frac{1}{\lvert y_w\rvert}\log\pi_\theta(y_w\lvert x)\right)+\exp\left(\frac{1}{\lvert y_l\rvert}\log\pi_\theta(y_l\lvert x)\right)$
ORPO [29]	$-\log p_\theta(y_w\lvert x)-\lambda\log\sigma\left(\log\frac{p_\theta(y_w\lvert x)}{1-p_\theta(y_w\lvert x)}-\log\frac{p_\theta(y_l\lvert x)}{1-p_\theta(y_l\lvert x)}\right)$, where $p_\theta(y\lvert x)=\exp\left(\frac{1}{\lvert y\rvert}\log\pi_\theta(y\lvert x)\right)$

RLHF and DPO. Reinforcement Learning from Human Feedback (RLHF) [5] use unpaired data generated by the policy $\pi_\theta(y|x)$. It add a KL divergence penalty to the reward to restrict the policy distribution to change too much. The objective is to maximize the accumulated rewards:

$$L(\theta) = -\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(x,y)}[r(x,y) - \beta\mathbb{D}_{KL}(\pi_\theta(y|x)||\pi_{\text{ref}}(y|x)]. \tag{16}$$

Based on the RLHF formation, Direct Preference Optimization (DPO) [26] derives a implicit reward related to the policy:

$$r_{\text{DPO}}(x,y) = \beta\log\frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} + \beta\log Z(x), \tag{17}$$

where $Z(x) = \sum_y \pi_{\text{ref}}(y|x)exp(\frac{1}{\beta}r(x,y))$ is the partition function that is hard to estimate. DPO leverage BT loss to remove $Z(x)$ from the objective, resulting in the folllowing objective:

$$L_{\text{DPO}} = -\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(x,y)}[\log\sigma(\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta\log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)})]. \tag{18}$$

SimPO, SimPER and ORPO. Simple Preference Optimization (SimPO) [27] introduce a reward margin and use the average log probability of a sequence as the implicit reward to eliminates the need for a reference model in DPO:

$$L_{\text{SimPO}} = -\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(x,y)}[\log\sigma(\frac{\beta}{|y_w|}\log\pi_\theta(y_w|x) - \frac{\beta}{|y_l|}\log\pi_\theta(y_l|x) - \gamma)]. \tag{19}$$

Simple equationment with Perplexity optimization (SimPER) [28] utilize perplexity [55] and changes the form of BT loss into a simpler one:

$$L_{\text{SimPER}} = -\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(x,y)}[\exp(\frac{1}{|y_w|}\log\pi_\theta(y_w|x)) + \exp(\frac{1}{|y_l|}\log\pi_\theta(y_l|x))]. \tag{20}$$

Odds Ratio Preference Optimization (ORPO) [29] incorporates an odds ratio-based penalty in BT loss form into the conventional negative log-likelihood (NLL) loss:

$$L_{\text{ORPO}} = -\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(x,y)}[\log p_\theta(y_w|x) - \lambda\log\sigma(\log\frac{p_\theta(y_w|x)}{1-p_\theta(y_w|x)} - \log\frac{p_\theta(y_l|x)}{1-p_\theta(y_l|x)})], \tag{21}$$

where $p_\theta(y|x) = \exp(\frac{1}{|y|}\log\pi_\theta(y|x))$.

We will leverage the definition of $p_\theta(y|x)$ for further derivation.

D Analysis of Convergence Behaviors of Preference Optimization Methods

We aim at finding an ideal method of preference optimization that can converge quickly at a good point in the context of length control. We begin with rewriting the objectives into the form of BT loss. And then we analyze the convergence conditions of different optimization objectives based on practical assumptions.

Background assumptions from practice. For sequences with length $|y| > 1000$, the typical per-token log-probabilities lie in the range $\log p(y_t) \in [-5, 0]$. Let m be the point where the sigmoid function saturates. Let $\sigma(m) > 0.99$, then we have $m > 5$. We make a assumption about the training dynamics: after updating, a convergence of the model on preference satisfies $p_\theta(y_w \mid x) > p_\theta(y_l \mid x)$, where y_w and y_l denote the preferred and less-preferred responses, respectively. Note that This could lead to over-fitting if it holds for every sample. However, this assumption is reasonable when considering the average case over the input distribution $\mathcal{X}$ (e.g. *it is satisfied for a set of inputs with high probability mass*).

SFT. With some algebra, we can derive the BT form of SFT loss as:

$$L_{\text{SFT}} = -\log \sigma(\log \frac{p_\theta(y_w|x)}{1 - p_\theta(y_w|x)}), \tag{22}$$

where the penalty for the rejected responses is absent, or $p_\theta(y_l|x)$ is default to 0.5. Convergence achieved when:

$$\frac{p_\theta(y_w|x)}{1 - p_\theta(y_w|x)} > e^m \Rightarrow p_\theta(y_w|x) > \frac{e^m}{1 + e^m}. \tag{23}$$

For $m = 5(\sigma \approx 0.993)$, requires $p_\theta(y_w|x) > 0.993$. For sequences longer than 1000, cumulative probability requires:

$$\prod_{t=1}^{1000} p(y_w^{(t)}|x, y_w^1, ..., y_w^{t-1}) > 0.993. \tag{24}$$

This is fundamentally incompatible with typical token probabilities ($p \approx 0.05 - 0.5$). So SFT cannot effectively enforce preference equationment, which equations with intuition. In another word, the convergence of SFT depends completely on the input x. In the context of length control, fitting the model into limited generation modes leads to potential catastrophic forgetting.

ORPO For ORPO, it is complex due to the additional NLL loss.

$$L_{\text{ORPO}} = -\log \sigma[\log(\frac{1 + (1 + e^{-z})p_\theta(y_w|x)^{1/\lambda}}{e^{-z} - (1 + e^{-z})p_\theta(y_w|x)^{1/\lambda}})], \tag{25}$$

where $z = \log \frac{p_\theta(y_w|x)}{1-p_\theta(y_w|x)} - \log \frac{p_\theta(y_l|x)}{1-p_\theta(y_l|x)}$. Then we have:

$$2p_\theta(y_w|x)^{1/\lambda} > \frac{p_\theta(y_l|x) - p_\theta(y_w|x)}{p_\theta(y_l|x) + p_\theta(y_w|x)} + m \tag{26}$$

Continue the derivation:

$$\log(1 + (1 + \exp(-z))p_\theta(y_w|x)^{\frac{1}{\lambda}}) - \log(\exp(-z) - (1 + \exp(-z))p_\theta(y_w|x)^{\frac{1}{\lambda}}) > m \tag{27}$$

$$1 + (1 + \exp(-z))p_\theta(y_w|x)^{\frac{1}{\lambda}} > \exp(m - z) - \exp(m)(1 + \exp(-z))p_\theta(y_w|x)^{\frac{1}{\lambda}} \tag{28}$$

$$2p_\theta(y_w|x)^{\frac{1}{\lambda}} > \frac{\exp(m - z) - 1}{(1 + \exp(m - z))(1 + \exp(m))} = \frac{1}{1 + \exp(m)}(1 - \frac{2}{1 + \exp(m - z)}) \tag{29}$$

$$2p_\theta(y_w|x)^{\frac{1}{\lambda}} > \frac{1}{1 + \exp(m)} \cdot (1 - 2\sigma(z - m)) \tag{30}$$

Note that z is inside the sigmoid function of the BT loss from the original objective. Then $z - m$ controls whether the right side is positive. If $z - m > 0$ (e.g, $p_\theta(y_w|x) > p_\theta(y_l|x)$), the right side is negative, which makes the inequality hold. Then the convergence depends on the SFT loss from the original objective. Then we can apply the analysis of SFT. If $z - m < 0$ (e.g, $p_\theta(y_w|x) < p_\theta(y_l|x)$), then we have

$$\frac{\text{sf}}{1 + \exp(m)}^{\frac{1}{\lambda}} < p_\theta(y_w|x) < p_\theta(y_l|x), \tag{31}$$

where sf is a positive scaling factor smaller than 1. Based on the assumption that $p_\theta(y_w|x) > p_\theta(y_l|x)$ *is satisfied for a set of inputs with high probability mass*. We consider the latter case as under-fit. In conclusion, ORPO fits better and faster to the preference data than SFT. **However, due to the NLL part of its objective, it fits worse than pure log odds ratio loss.**

SimPER SimPER cannot be written into a log sigmoid formation. Because the value of log sigmoid function always falls in $(-\infty, 0)$, while the inverse objective of SimPER is always positive with the assumption that $p_\theta(y_l|x) < p_\theta(y_w|x)$ *is satisfied for a set of inputs with high probability mass*. This implies that the convergence of SIMPER is unconstrained with preference. From the formulation of SIMPER's objective function, it can be observed that it separately optimizes two objectives of the following form:

$$p_\theta(y|x) = \exp\left(\frac{1}{|y|}\log \pi_\theta(y|x)\right) > 0, \tag{32}$$

which cannot be wriiten into a log sigmoid formation either. Thus, the convergence of SIMPER depends on the both responses.

DPO and SimPO These two methods have similar objective. We can directly extract the difference of the rewards from their objectives. For DPO we have:

$$\log \pi_\theta(y_w|x) - \log \pi_\theta(y_l|x) > \log \frac{\pi_{\text{ref}}(y_w|x)}{\pi_{\text{ref}}(y_l|x)} + \frac{m}{\beta} \tag{33}$$

For SimPO we have:

$$\frac{1}{|y_w|}\log \pi_\theta(y_w|x) - \frac{1}{|y_l|}\log \pi_\theta(y_l|x) > \frac{\gamma + m}{\beta} \tag{34}$$

Intuitively, the normalized advantage objective in SimPO potentially provides superior convergence properties for long-form generation tasks, while DPO remains valuable for maintaining response diversity. The convergence behaviors depend on the hyper-parameters settings, which decide the right parts of the equation. Note that the left parts can be seen as the reward differences of DPO and SIMPO. Thus, the right parts indicate reward margins floor.

Conclusion From the performance and analysis of SFT, ORPO and SIMPER, we can conclude that Bradley-Terry loss provide a predictable convergence characteristic from the objective, while NLL loss potentially amplifies the impact of the data on convergence. Under the form of BT loss, the tunable part of the objective is the reward function. Thus, a negative reward is needed to mitigate the effect of NLL. Besides, from the performance and anlysis of DPO and SimPO, we can conclude that the reward margin is important in length control, which is reflected directly in the affact caused by the hyper-parameters.

E Broader Impact

Our work proposed a post-training technique, which has no direct social impact but can be potentially used for tuning a large language model without supervision from the research community and society.