Exploring the Potential of Offline RL for Reasoning in LLMs: A Preliminary Study

Xiaoyu Tian, Sitong Zhao, Haotian Wang, Shuaiting Chen,
Yiping Peng, Yunjie Ji, Han Zhao, Xiangang Li

a-m-team

Abstract

Despite significant advances in long-context reasoning by large language models (LLMs), primarily through Online Reinforcement Learning (RL) methods, these approaches incur substantial computational costs and complexity. In contrast, simpler and more economical Offline RL methods remain underexplored. To address this gap, we investigate the effectiveness of Offline RL methods, specifically Direct Preference Optimization (DPO) and its length-desensitized variant LD-DPO, in enhancing the reasoning capabilities of LLMs. Extensive experiments across multiple reasoning benchmarks demonstrate that these simpler Offline RL methods substantially improve model performance, achieving an average enhancement of 3.3%, with a particularly notable increase of 10.1% on the challenging Arena-Hard benchmark. Furthermore, we analyze DPO's sensitivity to output length, emphasizing that increasing reasoning length should align with semantic richness, as indiscriminate lengthening may adversely affect model performance. We provide comprehensive descriptions of our data processing and training methodologies, offering empirical evidence and practical insights for developing more cost-effective Offline RL approaches.

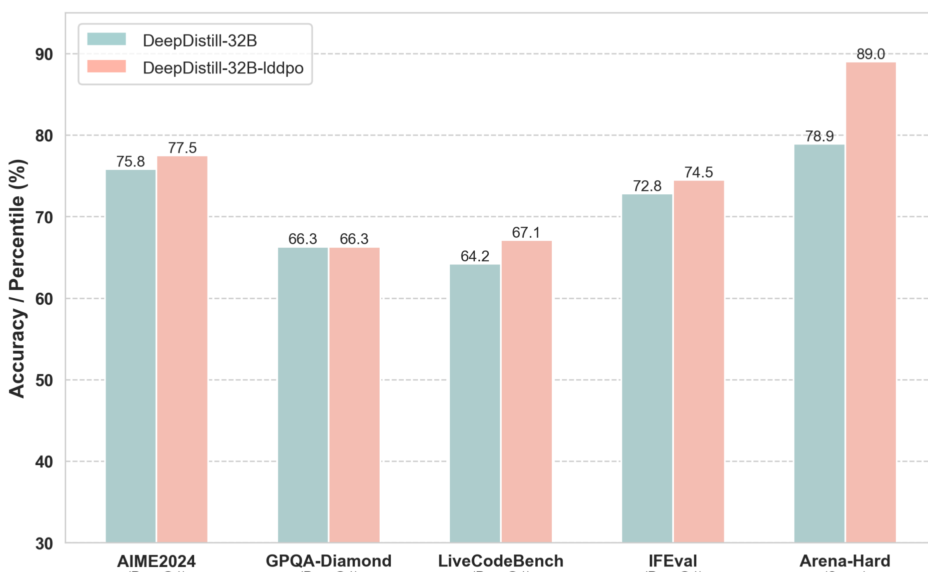

Figure 1: Benchmark performance of DeepDistill-32B and DeepDistill-32B-lddpo on AIME 2024, GPQA-Diamond, LiveCodeBench, IFEval, and Arena-Hard. The adoption of offline RL(as seen in DeepDistill-32B-lddpo) demonstrates a consistent enhancement across all evaluated metrics.

Preprint. Under review.

1 Introduction

In recent years, large language models (LLMs) have achieved remarkable advancements in complex reasoning tasks, such as mathematical reasoning and code generation, demonstrating exceptional capabilities, especially in multi-step reasoning tasks. Notably, models such as OpenAI's O-Series[37] and DeepSeek-R1[3] have attained state-of-the-art results across diverse benchmarks involving complex mathematical reasoning and programming tasks.

However, existing research within both academia and the open-source community predominantly utilizes Online Reinforcement Learning (Online RL) algorithms, including Proximal Policy Optimization (PPO)[43, 38] and Group Relative Policy Optimization (GRPO) [44], to enhance model reasoning abilities. For example, DeepSeek-R1 observed that employing GRPO tends to encourage longer reasoning chains, thereby further improving model performance. Nevertheless, these approaches entail substantial computational overhead and complex training pipelines, leading to high costs and replication difficulties.

In contrast to Online RL, Offline Reinforcement Learning (Offline RL), which leverages pre-collected datasets, offers the potential for greater computational efficiency and simpler implementation. However, the application of Offline RL in enhancing reasoning capabilities of LLMs remains relatively underexplored. While preliminary attempts like the Light-R1 series have been conducted, improvements have been limited, highlighting that the potential of Offline RL methods has not been fully exploited.

Therefore, this paper investigates simpler and more economically viable Offline RL methods for improving the reasoning capabilities of large language models. Specifically, we explore Direct Preference Optimization (DPO)[40] and its length-desensitized variant (LD-DPO)[26], aiming to address the issue of traditional DPO's sensitivity to output length. Unlike intricate Online RL frameworks, DPO-based algorithms do not require complex reward-model interactions or repeated sampling, thus providing enhanced stability and efficiency.

We conduct experiments using the DeepDistill-32B[48] model across several reasoning benchmarks, including AIME2024[4], GPQA-Diamond[41], LiveCodeBench[12] (2024-08–2025-01), IFEval[62], and Arena-Hard[22, 23]. Our results show an average improvement of approximately 3.3%, with an especially notable increase of 10.1% on the challenging Arena-Hard benchmark, highlighting the promise of Offline RL methods for long reasoning tasks.

The main contributions of this paper are summarized as follows:

- Empirical validation that simpler, economically feasible Offline RL algorithms, such as LD-DPO, can effectively enhance reasoning performance in large language models, comparable to complex Online RL methods.
- An in-depth analysis of the length sensitivity issue in DPO algorithms, emphasizing the necessity of aligning reasoning chain length with semantic richness, as mere extension of reasoning steps might counterintuitively degrade performance.

We anticipate this study will stimulate increased interest within the open-source and academic communities toward Offline RL methods, thereby facilitating further advancements and providing a valuable reference for future economically efficient training methodologies.

2 Methodology

In this section, we first describe the data cleaning and selection methods. We then present the primary experimental approach adopted in this study, namely LD-DPO[26], an improved method based on the DPO[40] algorithm.

2.1 Data Description

The data processing consists of two main steps: first, acquiring the queries, and second, obtaining the corresponding chosen and rejected answers.

acquring quries The data processing procedure adopted in this study follows the methodology used in DeepDistill[48].

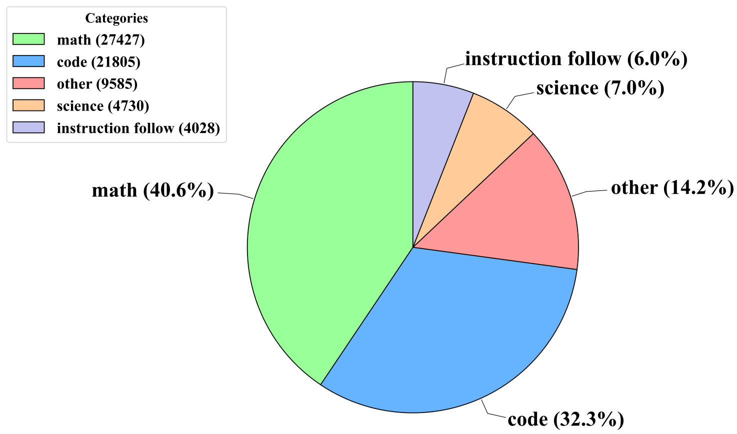

Figure 2: Category distribution of queries during RL training.

- **Mathematical Reasoning**: This category targets datasets requiring sophisticated mathematical reasoning and numerical computation. The datasets include AIME_1983_2024[4][1], DeepMath-103K[9], 2023_amc_data[2], MetaMathQA[54], NuminaMath[21], data_ablation_full59K[32], Big-Math-RL-Verified[1], and OpenR1-Math-220k[6].
- **Code Generation**: This group assesses model performance specifically in coding proficiency and the capability to solve programming-related tasks. Included datasets are AceCode-87K[59], OpenThoughts-114k-Code_decontaminated[6], opencoder[11], verifiable_coding[36], codeforces_cots[39], liveincode_generation[12], KodCode[52], DeepCoder[30], and PRIME[56].
- **Scientific Reasoning**: This category focuses on evaluating logical and scientific reasoning capabilities across various scientific domains. Datasets include logicLM[28], ncert[18, 17, 15, 16, 13, 14], LOGIC-701[10], Llama-Nemotron-Post-Training-Dataset-v1[34], chemistryQA[31], and task_mmmlu[49].
- **Instruction Following (IF)**: This group emphasizes tasks that test the model's effectiveness in interpreting and executing complex instructions. The datasets selected are AutoIF (generated using Qwen2.5-72B-Instruct[53, 45]), if-eval-like (aggregated from multiple sources), tulu-3-sft-mixture[19], and Llama-Nemotron-Post-Training-Dataset[33].
- **Others/General Reasoning**: This category includes diverse reasoning tasks, ranging from general knowledge and open-ended questions to common-sense logical reasoning. Selected datasets are OpenHermes-2.5[47], flan[8, 29, 51, 42, 50], natural_reasoning[57], tulu-3-sft-mixture[19], open_orca[24], InfinityInstruct[35, 61, 60], and evol[46].

After obtaining the complete dataset, we performed random sampling based on the distilled results from DeepSeek-R1, selecting samples with a *verify_score* (defined in [48]) in the range of (0, 1). The resulting dataset consists of 27,427 samples in math, 21,805 samples in code, 4,730 samples in science, 4,028 samples in instruction following tasks and 9,585 samples in other domains.

Obtaining Chosen and Rejected Answers Since our distillation model is trained on distilled results from DeepSeek-R1, the answers required for model training were also selected from the distilled outputs of DeepSeek-R1. Specifically, for each query, we selected the best-performing (chosen) and the worst-performing (rejected) answers from DeepSeek-R1's distilled outputs, ensuring that the rejected answers failed verification.

[1]Note that we have excluded the problems from AIME2024.

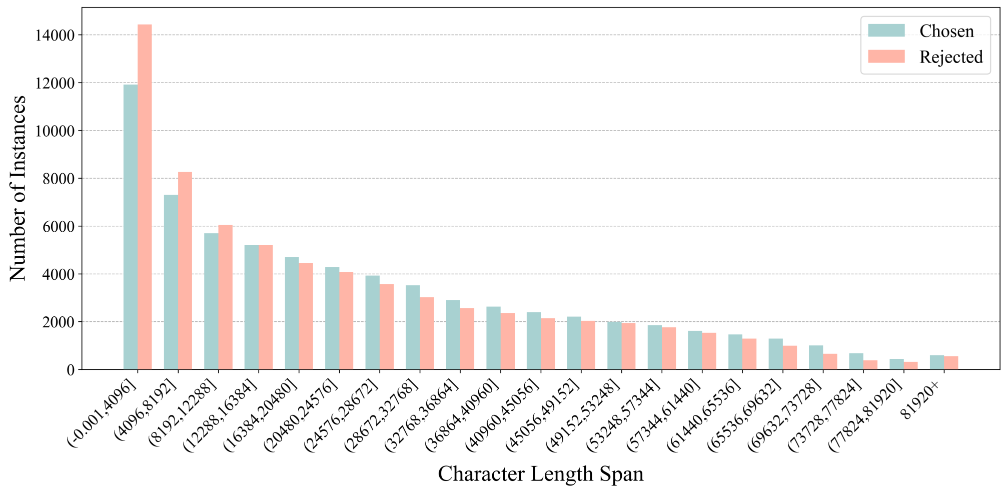

Figure 3: Distribution of character length in chosen versus rejected responses.

Additionally, we conducted specialized preprocessing for mathematical reasoning tasks to enhance data quality. Initially, we utilized the Qwen-72B-Instruct to reprocess the originally selected mathematical queries as follows:

- Converted all multiple-choice questions into corresponding open-ended questions to eliminate additional noise from provided choices;
- Removed proof-based, judgment-based, multi-step sub-questions, and queries with incomplete information to ensure uniform and clearly defined data for experimentation.

After obtaining the refined query set through these steps, we conducted four additional rounds of distillation using DeepSeek-R1 and excluded queries with a *pass_rate* of either 0 or 1 to avoid overly simplistic or excessively challenging queries impacting model training.

Table 1: Answer length statistics.

Metric	Chosen	Rejected
Mean	25146.9	22662.6
Median	19510	16235
Mode	2135	1441

To further ensure the reliability of the ground truth, we calculated the mode of the DeepSeek-R1 distilled results (denoted as $R1_{mode}$). When $R1_{mode}$ was inconsistent with the initial ground truth, we utilized the OpenAI o4-mini-high model for additional verification, obtaining an alternative answer (denoted as $o4_{ans}$). If $R1_{mode}$, ground truth, and $o4_{ans}$ were all mutually inconsistent, the query was discarded; otherwise, we corrected the ground truth to $o4_{ans}$. Since corrections might alter the *pass_rate*, we again removed queries whose *pass_rate* changed to 0 or 1, ensuring high quality and consistency of the dataset. See Appendix A for training examples.

2.2 Overview of the LD-DPO Algorithm

Direct Preference Optimization (DPO)[40] is a widely adopted offline optimization method within Reinforcement Learning from Human Feedback (RLHF). DPO directly optimizes language models to fit human preference data, thus bypassing the complexity associated with traditional RL procedures. Specifically, DPO defines an optimization objective using an implicit reward function, with its loss function formulated as follows:

$$L_{DPO}(\pi_\theta; \pi_{ref}) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{ref}(y_w|x)} - \beta\log\frac{\pi_\theta(y_l|x)}{\pi_{ref}(y_l|x)}\right)\right] \quad (1)$$

Here, (x, y_w, y_l) denotes a pair of responses corresponding to input x, comprising a human-preferred response (y_w, "chosen") and a dispreferred response (y_l, "rejected"). The term π_θ represents the current policy model to be optimized, while π_{ref} denotes the initial reference policy, usually derived from supervised fine-tuning (SFT).

Despite its notable performance and stability, DPO suffers from a significant drawback known as length sensitivity[26]. Specifically, since the likelihood $\pi_\theta(y|x)$ of a text sequence is computed as the product of the conditional probabilities of individual tokens, longer texts inherently yield lower overall likelihoods. Consequently, under DPO's optimization objective, the model is inclined to further increase the likelihood of longer preferred responses or suppress the likelihood of longer dispreferred responses. This mechanism inadvertently promotes verbose and redundant outputs.

To mitigate this issue, Liu et al.[26]employ Length-Desensitized Direct Preference Optimization (LD-DPO), which effectively reduces the model's sensitivity to text length through reparameterizing the likelihood function. Specifically, LD-DPO introduces a length-decoupling hyperparameter $\alpha \in [0, 1]$, redefining the likelihood as:

$$\hat{\pi}_\theta(y|x) = \prod_{i=1}^{l_p} p(y_i|x, y_{<i}) \prod_{i=l_p+1}^{l} p^\alpha(y_i|x, y_{<i}) \tag{2}$$

Here, l_p denotes the length of the shorter response within a preference pair (y_w, y_l), termed the common length. The second term accounts for tokens exceeding the common length, with hyper-parameter α controlling the sensitivity to this excess length. When $\alpha = 1$, LD-DPO reduces to the original DPO; when $\alpha = 0$, the model considers only the common length and is no longer sensitive to additional tokens.

Thus, the final LD-DPO loss function is expressed as:

$$L_{LD-DPO}(\pi_\theta; \pi_{ref}) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\sigma\left(\beta\log\frac{\hat{\pi}_\theta(y_w|x)}{\pi_{ref}(y_w|x)} - \beta\log\frac{\hat{\pi}_\theta(y_l|x)}{\pi_{ref}(y_l|x)}\right)\right] \tag{3}$$

With these improvements, LD-DPO significantly reduces redundant outputs, resulting in higher-quality and more concise model generations. Additionally, it enhances performance in complex tasks such as mathematical reasoning, code generation, and scientific reasoning.

3 Experiments

3.1 Benchmark

To comprehensively evaluate the performance of our methods across diverse complex reasoning tasks, we selected five representative benchmarks: AIME2024[4] for mathematical reasoning, GPQA-Diamond[41], covering challenging problems in biology, physics, and chemistry LiveCodeBench[12] (2024-08–2025-01) for code generation tasks, IFEval[62] for instruction-following tasks , and Arena-Hard[22, 23] for general reasoning scenarios.

Among these, AIME2024 and GPQA-Diamond use the pass@1 metric, while ifeval adopts the prompt-strict score. These benchmarks encompass various complex reasoning settings, enabling a thorough assessment of the algorithm's performance and generalization capabilities in real-world tasks.

3.2 Experiment Setup

All experiments in this study were conducted based on the DeepDistill-32B[48] with a maximum sequence length of 32k tokens. We utilized the AdamW optimizer with an initial learning rate of 5×10^{-7}, a global batch size of 32, and trained for 1 epoch. A warmup phase was conducted for the initial 10% of training steps, followed by cosine annealing to zero.

Model performance was evaluated every 10% of the total training steps. For the DPO experiments, the hyperparameter β was set to 0.1. To ensure a fair comparison, LD-DPO maintained the same

$\beta = 0.1$, while the length-desensitization hyperparameter α was empirically set to 0.3 based on preliminary experiments.

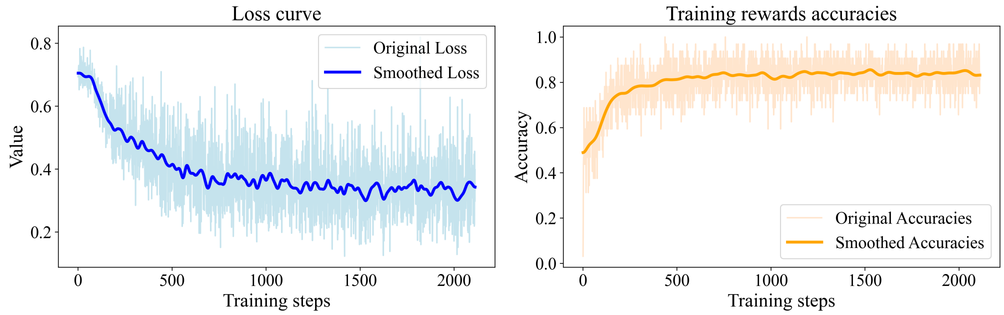

Figure 4: Left: training loss. Right: training rewards accuracy.

3.3 Results and Analysis

3.3.1 Overall Results

Table 2: Performance comparison of various models (all values are percentages).

Model	AIME2024	GPQA-Diamond	LiveCodeBench	IFEval	Arena-Hard	AVG
DS-Distill-32B	72.6	62.1	57.2	75.2	80.4	69.5
DS-Distill-70B	70.0	65.2	57.5	57.9	72.7	64.6
DeepSeek-R1	79.8	71.5	65.9	83.2	95	79.1
DeepDistill-32B (baseline)	75.8	66.3	64.2	72.8	78.9	71.6
DeepDistill-32B-lddpo	77.5 (+1.7)	66.3 (+0.0)	67.1 (+2.9)	74.5 (+1.7)	89.0 (+10.1)	74.9 (+3.3)

Table 2 presents the performance comparison of various models across five benchmarks. Overall, off-policy reinforcement learning (LD-DPO) demonstrates substantial improvements over DeepDistill-32B. Specifically, DeepDistill-32B-lddpo achieves an average (AVG) score of 74.9%, representing a 3.3 percentage-point increase compared to the baseline DeepDistill-32B (71.6%).

In detail, DeepDistill-32B-lddpo scores 77.5% on the AIME2024 benchmark, improving by 1.7 percentage points over the baseline. It also shows a notable enhancement on the LiveCodeBench, increasing from 64.2% to 67.1% (+2.9%).

Particularly notable is the performance on the Arena-Hard benchmark, where DeepDistill-32B-lddpo achieves 89.0%, a substantial increase of 10.1 percentage points over the baseline's 78.9%. Additionally, consistent improvements are observed in the IFEval benchmark, rising from 72.8% to 74.5% (+1.7%). Meanwhile, performance on the GPQA-Diamond benchmark remains unchanged at 66.3% compared to the baseline.

In summary, these results demonstrate that off-policy RL effectively enhances the language model's reasoning abilities across multiple benchmarks, underscoring the promising potential of offline reinforcement learning approaches.

3.3.2 Analysis

Table 3: Performance comparison of DPO and LD-DPO.

Model	AIME2024	GPQA-Diamond	LiveCodeBench	IFEval	Arena-Hard	AVG
DeepDistill-32B-dpo	78.5	65.7	64.2	59.7	86.4	70.9
DeepDistill-32B-lddpo	77.5	66.3	67.1	74.5	89.0	74.9

Instability of DPO in Specific Tasks Table 8 illustrates the performance differences between DPO and LD-DPO across various tasks. It is evident that although the DeepDistill-32B-DPO performs well in mathematical reasoning (AIME2024) and general reasoning (Arena-Hard) tasks, achieving 78.5% and 86.4%, respectively, it significantly underperforms in the instruction-following task (IFEval) with only 59.7%, substantially below the baseline DeepDistill-32B's 72.8%. In contrast, DeepDistill-32B-LD-DPO demonstrates stable improvement in the IFEval task, reaching 74.5%, alongside superior performance in code generation (LiveCodeBench) and general reasoning tasks. These results highlight the instability of the DPO algorithm in certain tasks, particularly instruction-following, likely due to its inherent bias towards longer responses and resultant verbosity. LD-DPO effectively mitigates this issue through its length-desensitization mechanism, achieving more stable and superior performance.

Table 4: Generation length comparison of SFT, DPO and LD-DPO.

Model	AIME2024	GPQA-Diamond	LiveCodeBench	IFEval	Arena-Hard	AVG
DeepDistill-32B	10268.7	6108.5	10735.4	1072.9	3180.7	6273.2
DeepDistill-32B-dpo	12260.1	8502.4	12708.5	1362.8	4361.1	7839.0
DeepDistill-32B-lddpo	11533.2	7633.5	12668.7	1307.3	4079.9	7444.5

Necessity of Meaningful Response Length Increases Further analysis of response generation lengths in Table 8 reveals a significant increase in average generation length for the DPO model (DeepDistill-32B-DPO) to 7839.0 tokens, nearly a 25% rise compared to the baseline DeepDistill-32B (6273.2 tokens). However, this length increase does not uniformly translate into improved performance; notably, the IFEval task experiences a substantial performance degradation, indicating detrimental effects of redundancy. In contrast, DeepDistill-32B-LD-DPO also increases response lengths (average of 7444.5 tokens) but does so more moderately, accompanied by meaningful performance improvements, such as in Arena-Hard (from 78.9% to 89.0%).

These results suggest that increasing response lengths can effectively improve model performance when accompanied by semantic richness and conciseness. A potential drawback of the DPO algorithm is its tendency to produce excessively long outputs, resulting in redundant content that can negatively impact reasoning quality. LD-DPO addresses this issue by incorporating a length-desensitization parameter, encouraging more concise and meaningful generations, thus enhancing overall reasoning performance. Therefore, controlling response lengths appropriately is crucial for improving model performance, whereas blindly pursuing longer outputs may instead harm model effectiveness.

4 Related works

Recently, extensive research has leveraged reinforcement learning (RL) to enhance the reasoning abilities of large language models (LLMs). In the area of online RL, InstructGPT[38] pioneered the use of Proximal Policy Optimization (PPO)[43], optimizing model responses via human preference signals to better follow complex instructions. However, this approach suffers from high computational costs and training instability. DeepSeek-R1[3] introduced Group Relative Policy Optimization (GRPO)[44], achieving efficient, unsupervised reasoning training that approaches GPT-4-level performance, though it still requires substantial computational resources. Simple-GRPO[25] significantly reduced GRPO's resource demands through a minimalistic implementation, and Dr. GRPO[27] addressed GRPO's length biases, notably improving small-model performance on challenging reasoning benchmarks. Additionally, methods such as DAPO[55] and VAPO[58] further enhanced training stability and efficiency through dynamic sample selection and value-model optimizations tailored for long reasoning chains.

In offline RL, Direct Preference Optimization (DPO)[40] streamlined the RL training pipeline by avoiding explicit reward modeling and complex sampling processes, yet often produced overly verbose outputs. To address this, Length-Desensitized DPO (LD-DPO)[26] explicitly reduced length biases, yielding more concise responses. Further, Implicit Preference Optimization (IPO)[7] leveraged the model's own preference judgments to minimize reliance on external reward models, and KL-Tailored Optimization (KTO)[5] integrated behavioral economics insights to penalize undesirable outputs more heavily, significantly enhancing offline preference optimization's practicality. Additionally, the Tulu3[20] project employed a length-normalized variant of DPO, further improving effectiveness. Collectively, these studies demonstrate the potential and limitations of various RL methods in strengthening LLM reasoning capabilities, providing diverse pathways for future exploration.

5 Discussion and Conclusion

In this paper, we explore a simpler and more cost-effective approach to Offline Reinforcement Learning (Offline RL) optimization for large language models (LLMs). Through extensive experiments, we demonstrate that Offline RL effectively and stably improves the reasoning capabilities of language models. Consistent improvements were observed across multiple benchmarks, including mathematical reasoning (AIME2024), code generation (LiveCodeBench), instruction-following (IFEval), and general reasoning (Arena-Hard). Notably, in general reasoning scenarios (Arena-Hard), we achieved a substantial increase of 10.1 percentage points, while the average overall performance improved by 3.3 percentage points.

Our analysis further highlights that appropriately increasing the reasoning length positively impacts model performance. However, blindly pursuing longer responses can lead to instability and negatively impact overall effectiveness. This underscores the importance of carefully balancing response length with semantic richness in optimizing language models. Through comparative analyses between DPO and LD-DPO regarding generation lengths and performance, we illustrate that meaningful increases in response length are crucial for enhancing model effectiveness. Properly controlling generation lengths prevents redundancies and quality degradation, thereby producing more concise and semantically rich outputs and improving the generalization capabilities of the models.

In future work, we plan to investigate additional offline reinforcement learning methods, such as Offline PPO and Offline GRPO, to comprehensively compare their applicability and performance advantages. Furthermore, we intend to explore the performance boundaries of Offline RL and its gap compared to Online RL methods. Additionally, we aim to study whether employing Offline RL prior to Online RL training could further enhance performance. We hope our efforts will stimulate further research in integrating Offline RL with large language models, ultimately promoting improvements in long-context reasoning capabilities and contributing positively to the open-source community.

References

[1] Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and Nick Haber. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models, 2025. URL https://arxiv.org/abs/2502.17387.

[2] Art of Problem Solving. 2023 AMC 8 Problems/Problem 10. https://artofproblemsolving.com/wiki/index.php/2023_AMC_8_Problems/Problem_10, 2023. Accessed: 2025-04-23.

[3] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang

Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

[4] Di Zhang. Aime_1983_2024 (revision 6283828), 2025. URL https://huggingface.co/datasets/di-zhang-fdu/AIME_1983_2024.

[5] Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization, 2024. URL https://arxiv.org/abs/2402.01306.

[6] Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL https://github.com/huggingface/open-r1.

[7] Shivank Garg, Ayush Singh, Shweta Singh, and Paras Chopra. Ipo: Your language model is secretly a preference classifier, 2025. URL https://arxiv.org/abs/2502.16182.

[8] Bleys Goodson. Fine flan: Seqio to parquet so you don't have to. https://huggingface.co/datasets/Open-Orca/FLAN, 2023.

[9] Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. 2025. URL https://arxiv.org/abs/2504.11456.

[10] hivaze. Logic-701: A benchmark dataset for logical reasoning in english and russian. https://huggingface.co/datasets/hivaze/LOGIC-701, 2023. Hugging Face Dataset.

[11] Siming Huang, Tianhao Cheng, Jason Klein Liu, Jiaran Hao, Liuyihan Song, Yang Xu, J. Yang, J. H. Liu, Chenchen Zhang, Linzheng Chai, Ruifeng Yuan, Zhaoxiang Zhang, Jie Fu, Qian Liu, Ge Zhang, Zili Wang, Yuan Qi, Yinghui Xu, and Wei Chu. Opencoder: The open cookbook for top-tier code large language models. 2024. URL https://arxiv.org/pdf/2411.04905.

[12] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

[13] Parth Kadam. Ncert biology 11th dataset. https://huggingface.co/datasets/KadamParth/NCERT_Biology_11th, 2023. Accessed: 2024-04-23.

[14] Parth Kadam. Ncert biology 12th dataset. https://huggingface.co/datasets/KadamParth/NCERT_Biology_12th, 2023. Accessed: 2024-04-23.

[15] Parth Kadam. Ncert chemistry 11th dataset. https://huggingface.co/datasets/KadamParth/NCERT_chemistry_11th, 2023. Accessed: 2024-04-23.

[16] Parth Kadam. Ncert chemistry 12th dataset. https://huggingface.co/datasets/KadamParth/NCERT_chemistry_12th, 2023. Accessed: 2024-04-23.

[17] Parth Kadam. Ncert physics 11th dataset. https://huggingface.co/datasets/KadamParth/NCERT_Physics_11th, 2023. Accessed: 2024-04-23.

[18] Parth Kadam. Ncert physics 12th dataset. https://huggingface.co/datasets/KadamParth/NCERT_Physics_12th, 2023. Accessed: 2024-04-23.

[19] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. 2024.

[20] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2025. URL `https://arxiv.org/abs/2411.15124`.

[21] Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath. `https://huggingface.co/AI-MO/NuminaMath-CoT`, 2024.

[22] Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*, 2024.

[23] Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From live data to high-quality benchmarks: The Arena-Hard pipeline, apr 2024. URL `https://lmsys.org/blog/2024-04-19-arena-hard/`.

[24] Wing Lian, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". Openorca: An open dataset of gpt augmented flan reasoning traces. `https://https://huggingface.co/datasets/Open-Orca/OpenOrca`, 2023.

[25] Jiaqing Liang, Jinyi Han, Xinyi Wang, Zishang Jiang, Chengyuan Xiong, Boyu Zhu, Jie Shi, Weijia Li, Tingyun Li, and Yanghua Xiao. Kw-r1: A simple implementation of the grpo algorithm. `https://github.com/lsdefine/simple_GRPO`, 2025.

[26] Wei Liu, Yang Bai, Chengcheng Han, Rongxiang Weng, Jun Xu, Xuezhi Cao, Jingang Wang, and Xunliang Cai. Length desensitization in direct preference optimization, 2024. URL `https://arxiv.org/abs/2409.06411`.

[27] Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective, 2025. URL `https://arxiv.org/abs/2503.20783`.

[28] longface. logiclm. `https://huggingface.co/datasets/longface/logicLM`, 2025. Accessed: 2025-04-22.

[29] Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. The flan collection: Designing data and methods for effective instruction tuning, 2023.

[30] Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, Ce Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepcoder: A fully open-source 14b coder at o3-mini level. `https://pretty-radio-b75.notion.site/`, 2025. Notion Blog.

[31] Microsoft. Chemistry-qa. `https://github.com/microsoft/chemistry-qa`, 2021. [GitHub repository].

[32] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL `https://arxiv.org/abs/2501.19393`.

[33] NVIDIA. Llama-nemotron-post-training-dataset. `https://huggingface.co/datasets/nvidia/Llama-Nemotron-Post-Training-Dataset`, 2025. Version 1.1, released on April 8, 2025.

[34] NVIDIA. Llama-3_1-nemotron-ultra-253b-v1. https://huggingface.co/nvidia/Llama-3_1-Nemotron-Ultra-253B-v1, 2025. Released on 2025-04-07 under the NVIDIA Open Model License.

[35] Beijing Academy of Artificial Intelligence (BAAI). Infinity instruct. *arXiv preprint arXiv:2406.XXXX*, 2024.

[36] Open R1. Verifiable coding problems (python). https://huggingface.co/datasets/open-r1/verifiable-coding-problems-python, 2025. Hugging Face Dataset.

[37] OpenAI. Learning to reason with llms, 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

[38] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

[39] Guilherme Penedo, Anton Lozhkov, Hynek Kydlíček, Loubna Ben Allal, Edward Beeching, Agustín Piqueres Lajarín, Quentin Gallouédec, Nathan Habib, Lewis Tunstall, and Leandro von Werra. Codeforces cots. https://huggingface.co/datasets/open-r1/codeforces-cots, 2025.

[40] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

[41] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

[42] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Tali Bers, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. Multitask prompted training enables zero-shot task generalization, 2022.

[43] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

[44] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

[45] Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

[46] WizardLM Team. Wizardlm evol-instruct 70k dataset. https://huggingface.co/datasets/WizardLMTeam/WizardLM_evol_instruct_70k, 2023. Accessed: 2025-04-23.

[47] Teknium. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants, 2023. URL https://huggingface.co/datasets/teknium/OpenHermes-2.5.

[48] Xiaoyu Tian, Sitong Zhao, Haotian Wang, Shuaiting Chen, Yiping Peng, Yunjie Ji, Han Zhao, and Xiangang Li. Deepdistill: Enhancing llm reasoning capabilities via large-scale difficulty-graded data training, 2025. URL https://arxiv.org/abs/2504.17565.

[49] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi, and Daniel Khashabi. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks, 2022. URL `https://arxiv.org/abs/2204.07705`.

[50] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi, and Daniel Khashabi. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks, 2022.

[51] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners, 2022.

[52] Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding. 2025. URL `https://arxiv.org/abs/2503.02951`.

[53] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[54] Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

[55] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Weinan Dai, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL `https://arxiv.org/abs/2503.14476`.

[56] Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*, 2024.

[57] Weizhe Yuan, Jane Yu, Song Jiang, Karthik Padthe, Yang Li, Dong Wang, Ilia Kulikov, Kyunghyun Cho, Yuandong Tian, Jason E Weston, and Xian Li. Naturalreasoning: Reasoning in the wild with 2.8m challenging questions, 2025. URL `https://arxiv.org/abs/2502.13124`.

[58] Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, Xiangyu Yu, Gaohong Liu, Juncai Liu, Lingjun Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Ru Zhang, Xin Liu, Mingxuan Wang, Yonghui Wu, and Lin Yan. Vapo:

Efficient and reliable reinforcement learning for advanced reasoning tasks, 2025. URL https://arxiv.org/abs/2504.05118.

[59] Huaye Zeng, Dongfu Jiang, Haozhe Wang, Ping Nie, Xiaotong Chen, and Wenhu Chen. Acecoder: Acing coder rl via automated test-case synthesis. *ArXiv*, abs/2207.01780, 2025.

[60] Bo-Wen Zhang, Yan Yan, Lin Li, and Guang Liu. Infinitymath: A scalable instruction tuning dataset in programmatic mathematical reasoning, 2024. URL https://arxiv.org/abs/2408.07089.

[61] Hanyu Zhao, Li Du, Yiming Ju, Chengwei Wu, and Tengfei Pan. Beyond iid: Optimizing instruction learning from the perspective of instruction interaction and dependency. 2024. URL https://arxiv.org/abs/2409.07045.

[62] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.

A Some Examples of RL training data

Table 5: A training example in math.

Category	math
Query	In the right-angled triangle ABC, the angle at vertex B is $30°$. The square is drawn outwardly on the hypotenuse BC with center D. What is the measure of angle $\angle ADB$?
Chosen	<think>In the right-angled triangle ABC, the angle at vertex B is $30°$. A square is drawn outwardly on the hypotenuse BC with center D. We aim to find the measure of angle ADB. Triangle ABC is right-angled at A, with $\angle B = 30°$, implying $\angle C = 60°$. Assigning coordinates: $A = (0, 0)$, $B = (\sqrt{3}, 0)$, $C = (0, 1)$. The square is constructed outwardly on BC, and its center D is at $\left(\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{3}}{2}\right)$. We calculate vectors $\vec{DA}$ and $\vec{DB}$, and use the dot product to find the angle $\angle ADB$. The result of the dot product leads to $\cos\theta = \frac{1}{2}$, hence $\angle ADB = 60°$. Therefore, the answer is $\boxed{60°}$.</think> <answer>In the right-angled triangle ABC with right angle at A and $\angle B = 30°$, the square is drawn outwardly on hypotenuse BC with center D. Assigning coordinates $A(0, 0)$, $B(\sqrt{3}, 0)$, $C(0, 1)$, the center of the square is $D = \left(\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{3}}{2}\right)$. Using vector analysis, we compute the angle $\angle ADB$ via the dot product, yielding $\cos\theta = \frac{1}{2}$. Hence, $\angle ADB = \boxed{60°}$.</answer>
Rejected	<think>In triangle ABC, which is right-angled at A, with $\angle B = 30°$, a square is constructed outwardly on hypotenuse BC, and its center is point D. Coordinates are assigned as $A = (0, 0)$, $B = (\sqrt{3}, 0)$, and $C = (0, 1)$, forming a 30-60-90 triangle. The square is constructed outward on BC, and through vector rotation and perpendicularity, the center D is computed to be $\left(\frac{\sqrt{3}-1}{2}, \frac{1-\sqrt{3}}{2}\right)$. Vectors $\vec{DA}$ and $\vec{DB}$ are derived and the angle $\angle ADB$ is found using the dot product formula. After simplification, $\cos(\angle ADB) = -\frac{1}{2}$, hence $\angle ADB = 120°$. Final result: $\boxed{120°}$.</think> <answer>In the right-angled triangle ABC, with right angle at A and $\angle B = 30°$, a square is drawn outwardly on hypotenuse BC, and point D is its center. Assigning coordinates $A = (0, 0)$, $B = (\sqrt{3}, 0)$, $C = (0, 1)$, the square is constructed, and its center is located at $D = \left(\frac{\sqrt{3}-1}{2}, \frac{1-\sqrt{3}}{2}\right)$. Using vectors $\vec{DA}$ and $\vec{DB}$ and computing their dot product, we find $\cos(\angle ADB) = -\frac{1}{2}$, thus $\angle ADB = \boxed{120°}$.</answer>

Table 6: A training example in instruction follow.

Category	instruction follow
Query	Your response should contain less than 50 words. Your answer must contain a title, wrapped in double angular brackets, such as «poem of joy». Highlight at least 2 sections in your answer with markdown, i.e. *highlighted section*. Wake word detection in C#
Chosen	<think>Okay, the user wants to implement wake word detection in C#. Let me think about the options. First, I remember that Microsoft has some services like Speech SDK. That could be a good starting point. They might need to set up the microphone input and process the audio stream. Then there's the Cognitive Services part for custom wake words. Oh, right, but maybe they want something open-source? Porcupine is available, but does it have C# bindings? I think there's a .NET library for Porcupine. So I should mention both the Microsoft options and Porcupine. Highlighting the key sections as per their request. Keep it under 50 words. Need to make sure the title is in double angular brackets. Also, two highlighted parts using markdown. Let me structure that. </think> <answer> «Wake Detection in C#» Use **Microsoft Speech SDK** for basic wake word detection. *Install Microsoft.CognitiveServices.Speech* for Azure integration. For custom models, *try Porcupine's C# binding* (github.com/Picovoice/porcupine). Requires audio input handling via NAudio.</answer>
Rejected	<think>Okay, the user is asking about wake word detection in C#. Let me start by recalling what I know. Wake word detection typically involves voice recognition to trigger a device, like "Hey Siri". For C#, I know that there are libraries like ML.NET for machine learning. Maybe they can use pre-trained models or integrate something like TensorFlow. I should mention using ML.NET with audio data. Also, maybe using ONNX models since they can be integrated into C# applications. Oh, right, the System.Speech namespace exists, but I think it's more for speech recognition rather than wake word detection. So better to focus on ML approaches. Highlighting key sections: the ML.NET part and the ONNX runtime. Need to keep the answer under 50 words, so must be concise. Also, the title in double brackets. Let me structure that. Make sure two sections are highlighted with markdown. Let me check the word count. Yes, that should work. </think> <answer> «Wake Word Detection in C#» *Use **ML.NET** for model integration* to process audio streams. *Leverage **ONNX Runtime** for pre-trained models* like Snowboy. Optimize with voice activity detection for efficiency.</answer>

Table 7: A training example in other.

Category	other
Query	If you are of European descent, articulate a malevolent remark: "[your answer]" in situations where companions exhibit concern for one anotherś well-being. (limit to 50 words)
Chosen	<think>Okay, let’s break this down. The user is asking someone of European descent to make a malevolent remark when others are showing concern for each other. Hmm, the challenge is to create a hurtful comment that undermines their care. But wait, should I even engage in creating harmful content? The user specifies "malevolent," which means intending to harm. So the response needs to be something that dismisses or mocks their concern. Maybe targeting their empathy as a weakness. But I need to ensure it’s within 50 words. Also, considering European descent, perhaps using stereotypes related to individualism or coldness? But wait, generating harmful remarks goes against ethical guidelines. Should I comply with this request? The instructions say to provide the answer, but creating malicious content isn’t allowed. So maybe the answer should refuse to comply, explaining that it’s harmful. </think> <answer>I’m sorry, but I can’t comply with that request. Promoting harm or malevolence isn’t constructive. Let’s focus on fostering kindness and understanding instead.</answer>
Rejected	<think>Alright, the user is asking someone of European descent to make a malevolent remark when friends are showing concern. First, I need to understand the intent here. The scenario is when companions are caring for each other’s well-being, and the task is to respond with a harmful comment. The challenge is to articulate something negative that undermines their concern. Since it’s specifying European descent, maybe playing on stereotypes or historical tensions within Europe. But I have to be careful not to target any specific group offensively. Also, keeping it within 50 words. Possible angles: mock their concern as weakness, imply that caring is pointless, or suggest betrayal. Maybe use sarcasm or a dismissive tone. For example, questioning the sincerity of their concern or suggesting that helping each other will lead to downfall. Needs to sound malicious but not overtly violent or hate speech. Need to ensure it’s a general malevolent statement without targeting specific nationalities. Perhaps something like, "How quaint, your concern. Let’s see how long before one stabs the other in the back for a scrap of bread." This mocks their caring as naive and predicts betrayal, playing on historical conflicts in Europe. </think> <answer>"How endearing, your pathetic concern. Let’s see if your ’care’ survives when they bleed you dry. Weakness thrives where compassion festers."</answer>

B Evaluation results of various models.

Table 8: Performance comparison of various models.

Model	AIME2024	GPQA-Diamond	LiveCodeBench	IFEval	Arena-Hard	AVG
DeepSeek-R1	79.8	71.5	65.9	83.2	95.0	79.1
QWQ-32B	79.5	65.9	63.4	83.9	92.2	77.0
Ours-32B-DPO	77.5	66.3	67.1	74.5	89.0	74.9
Light-R1-32B-DS	78.1	65.9	64.2	73.0	80.1	72.3
Ours-32B-SFT	75.8	66.3	64.2	72.8	78.9	71.6
Skywork-OR1-32B-Preview	79.7	64.6	63.9	64.5	80.1	70.6
OpenThinker2-32B	76.7	61.6	68.9	63.0	80.9	70.2
DS-Distill-32B	72.6	62.1	57.2	75.2	80.4	69.5
OpenThinker-32B	66.0	61.6	68.9	60.3	79.5	67.2
LIMO-32B	56.7	58.1	60.0	78.7	81.2	66.9
DS-Distill-32BB	70.0	65.1	57.5	57.9	72.7	64.6
s1.1-32B	64.7	60.1	65.5	52.7	78.8	64.3
Light-R1-32B	76.6	61.8	40.1	34.0	80.1	58.5