DAST: Difficulty-Aware Self-Training on Large Language Models

Anonymous ACL submission

Abstract

Present Large Language Models (LLM) self-training methods always under-sample on challenging queries, leading to inadequate learning on difficult problems which limits LLMs' ability. Therefore, this work proposes a difficulty-aware self-training (DAST) framework that focuses on improving both the quantity and quality of self-generated responses on challenging queries during self-training. DAST is specified in three components: 1) sampling-based difficulty level estimation, 2) difficulty-aware data augmentation, and 3) the self-training algorithm using SFT and DPO respectively. Experiments on mathematical tasks demonstrate the effectiveness and generalization of DAST, highlighting the critical role of difficulty-aware strategies in advancing LLM self-training.

1 Introduction

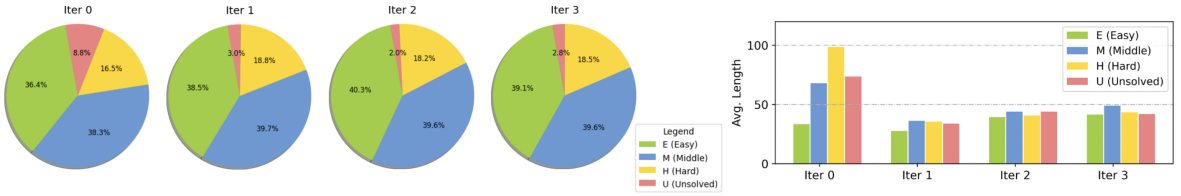
What doesn't kill you makes you stronger.
— Friedrich Wilhelm Nietzsche

The lack of extensive, high-quality human-curated training data for Large Language Models (LLMs) constrains the potential upper bounds of their capacities, particularly on complex reasoning tasks (Cobbe et al., 2021). Recently, self-training techniques of LLMs have garnered increasing attention, which iteratively fine-tunes LLMs on their self-generated outputs, attaining sustained improvements and diminishing the reliance on human interventions (Gulcehre et al., 2023; Singh et al., 2024; Huang et al., 2023; Zelikman et al., 2022).

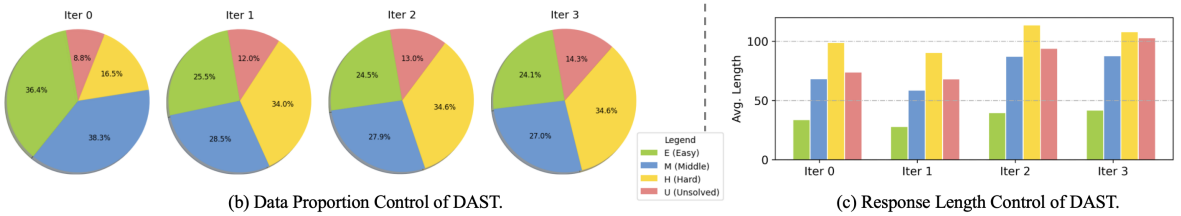
To ensure the quality of LLMs' self-generated training data, previous works employ rejection sampling (Sordoni et al., 2023) to filter out low-quality or incorrect responses with external reward models (Gulcehre et al., 2023) or ground-truth labels (Singh et al., 2024). This may lead to LLM over-sampling originally adept simple queries while under-sampling challenging queries (Ding et al., 2024; Tong et al., 2024). LLMs' insufficient learning in challenging instances is primarily in two aspects during self-training. First, when fixing the sampling number, only a few even or no correct responses are acquired on challenging queries, which iteratively exacerbates the distribution imbalance of the training data and severely overfitting on simple questions (Left hand of Figure 1 (a)). Second, the lengths of sampled self-generated responses on difficult questions are not enough (Right hand of Figure 1 (a)). Given that challenging problems require more thinking steps (Snell et al., 2024; Damani et al., 2024), the quality of these responses tends to be lower. As a result, LLMs can not adequately learn from challenging tasks, thereby restricting their capacity improvements.

Considering the above two issues, this work proposes a **d**ifficulty-**a**ware **s**elf-**t**raining (DAST) framework which focuses on increasing both the quantity and quality of self-generated responses on challenging queries during self-training: 1) DAST employs a sampling-based, model-specific method to estimate the difficulty level of each query. 2) Two data augmentation approaches are employed to balance the distribution and improve the response quality of training data given the difficulty levels. Specifically, we perform up-sampling on challenging questions to control the data proportion of different difficulty levels. We also employ a difficulty-matched few-shot prompting method to control the lengths of responses, encouraging LLMs to increase thinking steps on challenging questions. These two methods are combined incrementally. 3) We finally iteratively perform the above difficulty estimation and data augmentation steps in several rounds for LLM self-training using supervised fine-tuning (SFT) and direct preference optimization (DPO) (Rafailov et al., 2023) respectively.

Experiments are conducted on both the in-domain and out-of-domain tasks on various mathematical datasets. Results demonstrate that DAST



(a) Vanilla Self-Training Data Construction using Rejection Sampling.



(b) Data Proportion Control of DAST.

(c) Response Length Control of DAST.

Figure 1: Changes of data proportion and response length distribution of samples in different difficulty levels during a three-round self-training process. The vanilla rejection sampling to construct training data (a) is widely employed in Singh et al. (2024); Gulcehre et al. (2023); Sordoni et al. (2023); Zelikman et al. (2022). (b) and (c) are the proposed DAST aim to control data proportion and response lengths for challenging queries. Note that in iteration 0, the training data $\mathcal{D}_{\mathrm{u}}$ is the original dataset $\mathcal{D}_{\mathrm{o}}$ with ground-truth labels, while during iteration 1, 2, and 3, the training data is combined of self-generated data $\mathcal{D}_{\mathrm{a}}$ and the original dataset $\mathcal{D}_{\mathrm{o}}$. All the difficulty levels are measured on the initial policy $\mathcal{M}_0$ on the GSM8K test set and are fixed during self-training.

significantly enhances LLMs' math ability and generalizability over several baselines.

Our contributions are as follows: 1) This work first comprehensively incorporates difficulty level into LLM self-training, demonstrating the significance of difficulty for future works; 2) We propose two data augmentation methods in DAST to improve both quantity and quality on challenging queries using the estimated difficulty level; 3) We conduct experiments and validate that DAST can enhance LLM's math ability and generalizability using SFT and DPO respectively.

2 DAST Framework

2.1 Difficulty Level Estimation

We employ a sampling-based, model-specific method to estimate the difficulty level of each question to the model. Given the initial policy $\mathcal{M}_0$ and the training set $\mathcal{D}_{\mathrm{o}} = \{\boldsymbol{x}_i, \hat{\boldsymbol{r}}_i, \hat{\boldsymbol{y}}_i\}_{i=1}^{N}$, where $\boldsymbol{x}_i, \boldsymbol{r}_i, \boldsymbol{y}_i$ represent the question, rationale, and the ground-truth answer respectively. Each rationale $\hat{\boldsymbol{r}}_i = [\hat{r}_{i,1}, \ldots, \hat{r}_{i,l}]$ contains l reasoning steps where l varies in $\hat{\boldsymbol{r}}_i$. For each $(\boldsymbol{x}_i, \hat{\boldsymbol{r}}_i, \hat{\boldsymbol{y}}_i)$ and a prompt set $\mathcal{P}$ containing K different few-shot prompts, we employ each few-shot exemplar $\boldsymbol{p}_k \in \mathcal{P}$ with the question $\boldsymbol{x}_i$ for the policy $\mathcal{M}_0$ to generate the k-th response $\left(\boldsymbol{y}_i^{(k)}, \boldsymbol{r}_i^{(k)}\right) = \mathcal{M}_0\left(\boldsymbol{p}_k, \boldsymbol{x}_i\right)$ using temperature sampling ($T = 0.2, \text{top } p = 0.9$). We obtain the response set $\boldsymbol{Y}_i = \{\boldsymbol{y}_i^{(k)}\}_{k=1}^{K}$ and the label set $\boldsymbol{Z}_i = \left\{z_i^{(k)}\right\}_{k=1}^{K}$ by comparing each extracted answers in $\boldsymbol{Y}_i$ with the ground-truth $\hat{\boldsymbol{y}}_i$ to determine the correctness ($z_i^{(k)} \in \{0, 1\}$, 1 for *True* and 0 for *False*). The difficulty level d_i is estimated as follows [1]. Details and splits of four difficulty levels are in Table 3.

$$d_i = P\left(\boldsymbol{Y}_i | \boldsymbol{x}_i\right) = \frac{\sum_{k=1}^{K} \mathbb{I}\left(\boldsymbol{y}_i^{(k)} = \hat{\boldsymbol{y}}_i\right)}{K} \quad (1)$$

2.2 Data Augmentation

We augment $\mathcal{D}_{\mathrm{o}}$ with the strategy $\mathcal{A}(\cdot)$ for each query $\boldsymbol{x}_i$ according to d_i by controlling the data proportion and response lengths on $\mathcal{M}$ to obtain an augmented dataset $\mathcal{D}_{\mathrm{a}}$ for self-training as follows.

Data Proportion Control As in the left hand of Figure 1 (a), the construction of self-training data using rejection sampling may bias simple questions. Therefore, we set different sampling numbers K for different difficulty levels d_i of $\boldsymbol{x}_i$. More specifically, the sampling number K will multiply by a coefficient β determined by d_i as presented in Table 3. For $d_i \in \{M, H, U\}$ which indicates that $\boldsymbol{x}_i$ is a challenging question, β is larger to increase the number of correct responses sampled from the policy $\mathcal{M}$. The sampled responses will be added into $\mathcal{D}_{\mathrm{a}}$. As illustrated in Figure 1 (b), we can dynamically control the proportion of samples in all difficulty levels and balance the distribution of the training data in each self-training iteration.

[1] In this study, the challenging queries refer to the queries estimated in difficulty levels of Middle, Hard, and Unsolved

Algorithm 1 DAST Algorithm

Input: Training set $\mathcal{D}_{\text{o}}$, validation set $\mathcal{D}_{\text{v}}$, number of iterations $\mathcal{T}$, policy model at t-th iteration $\mathcal{M}_t$.
Output: Optimized policy π_θ.
for $t = 1$ to $\mathcal{T}$ **do**
  for $i = 1$ to $|\mathcal{D}_{\text{o}}|$ **do**
    Estimate difficulty level d_i of $\boldsymbol{x}_i$
    Obtain $\{\boldsymbol{r}_i^{(m)}, \boldsymbol{y}_i^{(m)}\}_{m=1}^{M} = \mathcal{A}(\boldsymbol{x}_i, d_i)$
    for $\boldsymbol{y}_i = \boldsymbol{y}_i^{(1)}$ to $\boldsymbol{y}_i^{(M)}$ **do**
      if $\boldsymbol{y}_i \equiv \hat{\boldsymbol{y}}_i$ **then**
        Label and add $(\boldsymbol{x}_i, \boldsymbol{r}_i^{+}, \boldsymbol{y}_i^{+})$ to $\mathcal{D}_a^{(t)}$
      else
        Label and add $(\boldsymbol{x}_i, \boldsymbol{r}_i^{-}, \boldsymbol{y}_i^{-})$ to $\mathcal{D}_a^{(t)}$
      end if
    end for
  end for
  Update training set $\mathcal{D}_{\text{u}} = \mathcal{D}_{\text{o}} \cup \mathcal{D}_a^{(t)}$
  while $\mathcal{M}_{t-1}$'s accuracy improves on $\mathcal{D}_v$ **do**
    Optimize $\mathcal{M}_{t-1}$ on $\mathcal{D}_{\text{u}}$ using SFT or DPO by minimizing $\mathcal{L}_{\text{sft}}/\mathcal{L}_{\text{dpo}}$ as in Equation 2 or 3
  end while
  $\mathcal{M}_t \leftarrow \mathcal{M}_{t-1}$
end for

Response Length Control As in the right hand of Figure 1 (a), the lengths of responses generated using the vanilla few-shot sampling method are in averaged length for all difficulty levels during self-training (iterations 1, 2, and 3) and relatively shorter than lengths of the ground-truth responses in $\mathcal{D}_{\text{o}}$ (iteration 0). To generate lengthy and difficulty-matched responses, we propose a difficulty-matched few-shot (DMFS) prompting method: for each difficulty level $d \in \{E, M, H, U\}$, we select samples from the training set that exceed the average response length of this difficulty level to construct four prompt sets $\mathcal{P}_E, \mathcal{P}_M.\mathcal{P}_H, \mathcal{P}_U$. DMFS examples are employed based on d_i to sample responses for $\boldsymbol{x}_i$ on $\mathcal{M}$. Sampled responses will be added into $\mathcal{D}_{\text{a}}$. Therefore, length distribution of $\mathcal{D}_{\text{a}}$ is close to the ground truth in iteration 0 as in Figure 1 (c), which improves the response quality with more thinking steps (Snell et al., 2024; Yeo et al., 2025).

2.3 Self-Training

As presented in Algorithm 1, in the t-th iteration, the training set $\mathcal{D}_{\text{u}}$ is updated by merging the augmented dataset $\mathcal{D}_{\text{a}}^{(t)}$ and initial training set $\mathcal{D}_{\text{t}}$, ensuring $\mathcal{D}_{\text{u}}$ doesn't diverge too much from $\mathcal{D}_{\text{t}}$. The policy $\mathcal{M}_j$ is fine-tuned based on $\mathcal{M}_{j-1}/\mathcal{M}_0$ on $\mathcal{D}_u$ using SFT/DPO (Rafailov et al., 2023) by optimizing $\mathcal{L}_{\text{sft}}/\mathcal{L}_{\text{dpo}}$ in Equation 2/3 respectively. $\mathcal{M}_j$ is trained to be converged while the accuracy doesn't increase on the validation set $\mathcal{D}_{\text{v}}$. Specifically, we denote DAST using SFT/DPO by **DAST-S/DAST-D**. For DAST-S, we investigate only employing data proportion control or length control, and denote by **DAST-P** and **DAST-L** respectively.

3 Experimental Setting

Datasets During the training stage, we jointly combine training sets from **GSM8K** (Cobbe et al., 2021) and **MATH** (Hendrycks et al., 2021) as $\mathcal{D}_{\text{t}}$. We evaluate **in-domain (ID)** performance on the corresponding test sets. We also assess the **out-of-domain (OOD)** performance three challenging test sets: **TAL-SCQ** (math eval, 2023) **College** (Tang et al., 2024), and **TheoremQA** (Chen et al., 2016). We standardize the data format as in Appendix E and employ the evaluation script of MWPBench [2] (Tang et al., 2024) to judge the correctness of the extracted answer compared with the ground-truth label. Dataset details are in Appendix C.

Baselines We utilize in-context learning (**ICL**) (Brown et al., 2020) to generate responses. We also employ several SFT-based and DPO-based baselines. SFT-based baselines include: 1) single-round standard **SFT** and difficulty-aware rejection tuning (**DART**) (Tong et al., 2024) (specified in **DART**-*Uniform* and **DART**-*Prob2Diff*); and 2) multi-round **ReST-EM** (Singh et al., 2024). DPO-based (Rafailov et al., 2023) baselines include single- and multi-round DPO (**DPO** and **mDPO**). Detailed implementations of the above baselines can be referred to Appendix D.

4 Results and Analysis

4.1 Main Experiments

Experiments are conducted on **Llama-3.1-8B** (Llama-3.1) (AI@Meta, 2024) in this work. As in Figure 2, several findings can be found below.

1. With different sizes of self-training data in each iteration, **DAST-S and DAST-D consistently yield superior performance over corresponding SFT and DPO baselines with comparable or less data**, exhibiting the effectiveness and efficiency of DAST for both SFT and DPO during self-training. Data size statistics are presented in Table 4.

2. DAST-P exhibits better performance compared to DAST-L, suggesting that **increasing the data size can gain more improvements than increasing the response lengths for challenging**

[2] https://github.com/microsoft/unilm/tree/master/mathscale

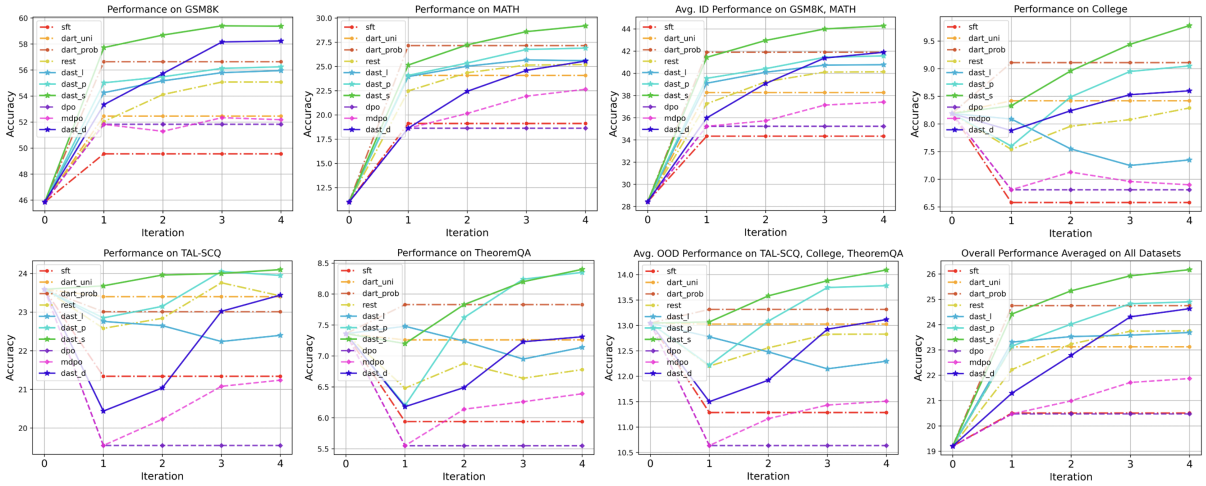


Figure 2: Performance results of DAST over various baselines on both in-domain (ID) and out-of-domain (OOD) mathematical test sets using Llama-3.1. Note that the names of employed baselines are in lowercase.

queries. This can be attributed to that the initial policy is suboptimal and the sampled lengthy responses are also low-quality. Therefore, raising the data quantity can lead to more obvious gains.

3. **DAST-S and DAST-P can better generalize to OOD tasks than others**. DAST enables LLMs to adequately learn more diverse challenging questions, thereby achieving more pronounced improvements in relatively challenging OOD tasks.

4.2 Effects of Data Proportion Control

In this part, we investigate the research question *"As self-training progresses iteratively, will increasing the proportion of challenging samples lead to further improvements?"*. We control the proportions of challenging queries with fixed data size in each iteration by adjusting β during self-training as illustrated in Figure 3. Results suggest that LLMs perform better when trained on the dataset with a balanced distribution (DAST-P-α1) of different difficulty levels than more hard samples (DAST-P-α2) during self-training. Excessive challenging samples may lead to a large distribution shift, affecting LLMs' original abilities on simple queries.

4.3 Effects of Response Length Control

In this part, we investigate the research question *"Will the performance be further improved by employing difficult examples across all queries to generate lengthy responses during self-training?"*. We generate training data using few-shot examples from solely a single difficulty level in the first round of DAST to compare with our proposed difficulty-matched few-shot (DMFS) prompting method for sampling. Results in Table 1 suggest that training

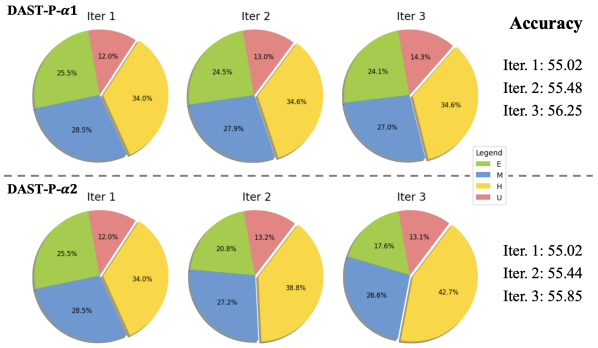


Figure 3: Results of data proportion control.

data generated by DMFS outperforms those obtained from any single level. Tailoring response length to difficulty levels of queries is more effective, as sampling lengthy responses to simple queries may result in overthinking and undermine performances (Halawi et al., 2024).

Exam. Level	E	M	H	U	DMFS
ID	35.58	37.44	38.90	38.66	**41.94**
OOD	11.45	12.15	12.48	12.06	**13.07**

Table 1: Results of response length control.

5 Conclusion

This work proposes a DAST framework to enhance both the quantity and quality of challenging queries during the self-training process, including three key parts: difficulty level estimation, data augmentation, and a self-training algorithm. Experiments conducted on math tasks using SFT and DPO showcase the effectiveness and generalization of DAST.

Limitations

The limitations of this work are as follows:

Response Quality This work enhances the response quality by solely increasing the thinking steps or lengths of responses. Although improving response quality by adding length is simple yet effective for challenging queries, more explorations should be conducted to comprehensively evaluate the response quality in other dimensions.

Task Expansion Another limitation is that the experiments are solely conducted on mathematical reasoning tasks. This constraint primarily arises from that many tasks like long-form generations are also challenging to evaluate the generation quality. Future research endeavors should prioritize a wider range of datasets of long-form generation tasks to thoroughly assess the applicability and effectiveness of DAST.

Acknowledgments

References

AI@Meta. 2024. Llama 3 model card. *AI@Meta*.

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

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. 2016. Training deep nets with sublinear memory cost. *Preprint*, arXiv:1604.06174.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Mehul Damani, Idan Shenfeld, Andi Peng, Andreea Bobu, and Jacob Andreas. 2024. Learning how hard to think: Input-adaptive allocation of lm computation. *Preprint*, arXiv:2410.04707.

Yiwen Ding, Zhiheng Xi, Wei He, Zhuoyuan Li, Yitao Zhai, Xiaowei Shi, Xunliang Cai, Tao Gui, Qi Zhang, and Xuanjing Huang. 2024. Mitigating tail narrowing in llm self-improvement via socratic-guided sampling. *arXiv preprint arXiv:2411.00750*.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. 2025. A survey on llm-as-a-judge. *Preprint*, arXiv:2411.15594.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. 2023. Reinforced self-training (rest) for language modeling. *Preprint*, arXiv:2308.08998.

Danny Halawi, Jean-Stanislas Denain, and Jacob Steinhardt. 2024. Overthinking the truth: Understanding how language models process false demonstrations. *Preprint*, arXiv:2307.09476.

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

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. 2024. V-star: Training verifiers for self-taught reasoners. *Preprint*, arXiv:2402.06457.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore. Association for Computational Linguistics.

Yiming Huang, Xiao Liu, Yeyun Gong, Zhibin Gou, Yelong Shen, Nan Duan, and Weizhu Chen. 2024. Key-point-driven data synthesis with its enhancement on mathematical reasoning. *Preprint*, arXiv:2403.02333.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Chengpeng Li, Zheng Yuan, Hongyi Yuan, Guanting Dong, Keming Lu, Jiancan Wu, Chuanqi Tan, Xiang Wang, and Chang Zhou. 2024. MuggleMath: Assessing the impact of query and response augmentation on math reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10230–10258, Bangkok, Thailand. Association for Computational Linguistics.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, and Dongmei Zhang. 2025. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *Preprint*, arXiv:2308.09583.

math eval. 2023. TAL-SCQ5K.

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

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, Abhishek Kumar, Alex Alemi, Alex Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Elsayed, Hanie Sedghi, Igor Mordatch, Isabelle Simpson, Izzeddin Gur, Jasper Snoek, Jeffrey Pennington, Jiri Hron, Kathleen Kenealy, Kevin Swersky, Kshiteej Mahajan, Laura Culp, Lechao Xiao, Maxwell L. Bileschi, Noah Constant, Roman Novak, Rosanne Liu, Tris Warkentin, Yundi Qian, Yamini Bansal, Ethan Dyer, Behnam Neyshabur, Jascha Sohl-Dickstein, and Noah Fiedel. 2024. Beyond human data: Scaling self-training for problem-solving with language models. *Preprint*, arXiv:2312.06585.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *Preprint*, arXiv:2408.03314.

Alessandro Sordoni, Eric Yuan, Marc-Alexandre Côté, Matheus Pereira, Adam Trischler, Ziang Xiao, Arian Hosseini, Friederike Niedtner, and Nicolas Le Roux. 2023. Joint prompt optimization of stacked llms using variational inference. *Advances in Neural Information Processing Systems*, 36:58128–58151.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. 2024. Mathscale: Scaling instruction tuning for mathematical reasoning. *Preprint*, arXiv:2403.02884.

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

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. 2024. DART-Math: Difficulty-Aware Rejection Tuning for Mathematical Problem-Solving. *arXiv preprint*. ArXiv:2407.13690 [cs].

Ke Wang, Jiahui Zhu, Minjie Ren, Zeming Liu, Shiwei Li, Zongye Zhang, Chenkai Zhang, Xiaoyu Wu, Qiqi Zhan, Qingjie Liu, and Yunhong Wang. 2024a. A survey on data synthesis and augmentation for large language models. *Preprint*, arXiv:2410.12896.

Tianduo Wang, Shichen Li, and Wei Lu. 2024b. Self-training with direct preference optimization improves chain-of-thought reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11917–11928, Bangkok, Thailand. Association for Computational Linguistics.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in llms. *Preprint*, arXiv:2502.03373.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

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

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *Preprint*, arXiv:2308.01825.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *Preprint*, arXiv:2203.14465.

Kun Zhou, Beichen Zhang, Jiapeng Wang, Zhipeng Chen, Wayne Xin Zhao, Jing Sha, Zhichao Sheng, Shijin Wang, and Ji-Rong Wen. 2024. Jiuzhang3.0: Efficiently improving mathematical reasoning by training small data synthesis models. *arXiv preprint arXiv:2405.14365*.

Notation	Description
$\mathcal{D}_{\text{o}}$	Training set containing N Question-Answering pairs. ($\lvert\mathcal{D}_{\text{o}}\rvert = N$)
$\mathcal{D}_{\text{v}}$	Validation set.
$\mathcal{P}$	Set of few-shot exemplars.
$\mathcal{M}_t$	Policy model in the t-th ieration where $\mathcal{M}_0$ is the initial policy.
$\boldsymbol{x}_i$	The i-th question sample.
$\hat{\boldsymbol{r}}_i$	The i-th ground-truth rationale path for $\boldsymbol{x}_i$.
$\boldsymbol{r}_i^{(k)}$	The k-th sampled rationale path to the i-th question $\boldsymbol{x}_i$.
$\hat{\boldsymbol{y}}_i$	The i-th ground-truth answer for $\boldsymbol{x}_i$.
$\boldsymbol{y}_i^{(k)}$	The k-th sampled response to the i-th question $\boldsymbol{x}_i$.
$\boldsymbol{p}_k$	k-th few-shot exemplar to sample $\boldsymbol{y}_i^{(k)}$.
K	Number of sampled responses.
$\boldsymbol{Y}_i$	Answering set containing K sampled response $\{\boldsymbol{y}_i^{(k)}\}$ for the i-th question $\boldsymbol{x}_i$.
$z_i^{(k)}$	The label of $\boldsymbol{y}_i^{(k)}$ ($z_i^{(k)} \in \{0, 1\}$, 1 for *True* and 0 for *False*).
$\boldsymbol{Z}_i$	Label set corresponding to $\boldsymbol{Y}_i$.
$\mathcal{L}_\alpha$	Training loss functions SFT or DPO where $\alpha \in \{\text{sft}, \text{dpo}\}$.
d_j	Estimated difficulty level for $\boldsymbol{x}$.
c	Co-efficient to control the data proportion of samples in different difficulty levels.
T	Temperature of sampling.
$\mathcal{T}$	Number of iterations.

Table 2: Summarized notations in this work.

A Protocols

A.1 Definition of Notations

The definitions of the notations in this work are summarized in Table 2.

A.2 Difficulty Level Split

p	Difficulty Level	Denotation d_j	β
[0.8, 1.0]	Easy	E	1
[0.4, 0.8)	Middle	M	3
(0.0, 0.4)	Hard	H	5
0.0	Unsolved	U	5

Table 3: Difficulty level split.

A.3 Equations

SFT is optimized by minimizing the negative log-likelihood loss as follows.

$$\mathcal{L}_{\text{sft}} = \mathbb{E}\left[-\log \mathcal{M}_{j-1}(\boldsymbol{y}_i^+, \boldsymbol{r}_i^+ | \boldsymbol{x})\right] \quad (2)$$

DPO is optimized to minimize the preference loss as follows.

$$\mathcal{L}_{\text{dpo}} = \mathbb{E}\left[-\log \sigma\left(\theta(\boldsymbol{y}_i^+, \boldsymbol{r}_i^+ | \boldsymbol{x}) - \theta(\boldsymbol{y}_i^-, \boldsymbol{r}_i^- | \boldsymbol{x})\right)\right] \quad (3)$$

where $(\boldsymbol{x}_i, \boldsymbol{y}_i^+, \boldsymbol{r}_i^+, \boldsymbol{y}_i^-, \boldsymbol{r}_i^-) \sim \mathcal{D}_{\text{u}}$ and $\theta(\cdot|\boldsymbol{x}) = \log \frac{\mathcal{M}_{j-1}(\cdot|\boldsymbol{x})}{\mathcal{M}_0(\cdot|\boldsymbol{x})}$.

B Related Works

LLM Self-Training LLM Self-Training (Gulcehre et al., 2023; Singh et al., 2024) involves a machine learning paradigm where a LLM iteratively improves its performance by generating and leveraging its own synthetic data for further training without human intervention also referring to self-taught (Zelikman et al., 2022; Hosseini et al., 2024), self-evolving (Tao et al., 2024), or self-improve (Huang et al., 2023). Such self-training paradigms always involve a generation step by prompting LLMs to self-generate training data and an improve step by training the LLM on the self-generated data (Gulcehre et al., 2023). In the Generation step, to ensure the data quality, the generated data are always filtered and selected using rejection sampling (Yuan et al., 2023) before being employed for training. These signals can be reward scores returned by a reward model (Gulcehre et al., 2023), the binary score to judge the correctness given gold answer for mathematical or coding tasks (Singh et al., 2024; Yuan et al., 2023; Zelikman et al., 2022; Wang et al., 2024b), or two scores using two reward model for process and object respectively on reasoning tasks (Yang et al., 2024). LLM itself can be also regarded as judge or the reward model (Yuan et al., 2024; Gu et al., 2025).

In the Improve step, the selected data are utilized to train the LLM using supervised fine-tuning (SFT) (Gulcehre et al., 2023; Zelikman et al., 2022;

Singh et al., 2024) or reinforcement learning (Gulcehre et al., 2023; Hosseini et al., 2024; Wang et al., 2024b). Some studies iteratively train the policy LLM based on the previously obtained LLM (Gulcehre et al., 2023) while some train the base LLM instead of the LLM obtained from the previous iteration (Wang et al., 2024b; Singh et al., 2024; Zelikman et al., 2022).

Data Synthesis on Math Problems Since the growth rate of high-quality data is significantly outpaced by the expansion of training datasets, synthetic data has emerged as a promising solution (Wang et al., 2024a) to address the data capacity limitation and further improve LLM performance according to scaling laws (Kaplan et al., 2020). Self-training paradigm employs LLM itself to generate the synthetic training data on mathematical problems (Singh et al., 2024; Zelikman et al., 2022; Wang et al., 2024b). Tong et al. (2024) proposes to synthesize more responses for challenging questions. Yu et al. (2024) bootstraps the diversity of math problems by re-writing the training set and further fine-tunes LLM on the enhanced training set. Li et al. (2024) designs several re-writing principles to enhance both questions and responses to obtain an enhanced training set. Luo et al. (2025) proposes to synthesize more complex and diverse mathematical instructions to improve LLMs' mathematical reasoning ability. Ding et al. (2024) employs the Socratic-Guided Sampling (GSI) method to synthesize data to address the long-tail distribution issue during self-training. Some studies also investigate to synthesizing new questions (Huang et al., 2024; Zhou et al., 2024)

C Dataset Details

GSM8K GSM8K (Cobbe et al., 2021) [3] is a high-quality multi-step mathematical reasoning dataset of diverse grade school math word problems constructed by human problem writers, including 7,472 training samples and 1,319 test samples. All the questions take 2 to 8 steps to solve, involving a series of basic arithmetic operations to parse the final answer.

MATH MATH (Hendrycks et al., 2021) [4] is a challenging mathematical dataset with competition mathematics problems, consisting of 7,500 training samples and 5,000 test samples. Each problem in MATH also has a full step-by-step solution which can be used to teach models to generate answer derivations and explanations across several subjects including algebra, geometry, number theory, counting and probability, calculus, etc.

TAL-SCQ TAL-SCQ5K-EN (math eval, 2023) [5] are high-quality mathematical competition datasets in English created by TAL Education Group with totally 5,000 samples. The TAL-SCQ dataset split 3,000 and 2,000 questions for training and testing respectively. The questions are in the form of multiple-choice and cover mathematical topics at different levels of primary, junior high, and high school. We format all the samples in standard QA format.

College (Tang et al., 2024) [6] The College dataset contains 1281 training and 2818 test college-level mathematical problems extracted from 9 textbooks across 7 domains such as linear algebra and differential equations. This dataset is to test generalization on complex mathematical reasoning in diverse domains.

TheoremQA (Chen et al., 2016) [7] The TheoremQA dataset contains 800 problems focused on utilizing mathematical theorems to solve challenging problems in fields such as math, physics, finance, and engineering, testing generalization on theoretical reasoning in general STEM. The dataset is collected by human experts with very high quality. We filter out the questions requiring pictures and remain 747 samples to test.

D Baseline Details

ReST-EM Sampling Stage: Set the sampling temperature to 0.5. For each query, sample 10 responses. Retain responses based on whether the final answer matches the ground truth. Training Stage: Combine the sampled data from the current policy model with the original dataset $\mathcal{D}_o$ to form a new training dataset, which is then used for supervised fine-tuning (SFT).

DAST-*Uniform* Sampling Stage: Set the sampling temperature to 0.5. During dataset construction, perform oversampling for difficult samples to ensure every sample has 4 correct responses. Training Stage: Combine the sampled data with the origi-

[3] https://github.com/openai/grade-school-math
[4] https://github.com/hendrycks/math/
[5] https://github.com/math-eval/TAL-SCQ5K
[6] https://github.com/microsoft/unilm/tree/master/mathscale/MWPBench
[7] https://github.com/wenhuchen/TheoremQA

nal dataset $\mathcal{D}_o$ to form a new training dataset, which is then used for supervised fine-tuning (SFT).

DAST-*Prob2Diff* Sampling Stage: Set the sampling temperature to 0.5. During dataset construction, perform oversampling for difficult samples, applying a coefficient based on the difficulty level. More challenging samples are assigned more responses. Training Stage: Combine the sampled data with the original dataset to form a new training dataset, which is then used for supervised fine-tuning (SFT).

DPO Sampling Stage: Set the sampling temperature to 0.5. The dataset construction is similar to SFT while we will also add negative samples into training data to conduct the DPO algorithm.

mDPO The sampling stage is similar to ReST-EM and we will also add negative samples into training data to conduct the DPO algorithm. For the multi-round DPO, we sample the self-generated training data on the model obtained from the previous training iteration but we train the model from the initial policy as in Equation 3.

E Prompt Template

Prompt and Problem Format

You are an excellent mathematician. Answer the following mathematical questions based on your knowledge.

Question ###: `{Question}`
Response ###:
<think>`{Reasoning steps}`</think>.
The answer is \box`{Answer}`.

F Implementation Details

Experiments are conducted on **Llama-3.1-8B** (Llama-3.1) [8] (AI@Meta, 2024).

During dataset construction, we sample the responses using 8-shot examples by setting the sampling temperature to $T = 0.5$. For response length control of DAST, challenging samples are paired with longer few-shot examples. When sampling, we will dynamically adjust the sampling number K to control the training data in each iteration comparable as in Table 4.

During training, ADAM parameter update is used in a mini-batch mode. The initial learning rate of 1e-4 is utilized with the 0.05 warm-up ratio and 0.01 weight decay of the ADAM optimizer.

Method	Iteration	Data Size
ICL	-	-
SFT	-	15k
DART-*Uniform*	-	60k
DART-*Prob2Diff*	-	60k
ReST-EM	1	50k
ReST-EM	2	55k
ReST-EM	3	58k
ReST-EM	4	58k
DAST-P	1	55k
DAST-P	2	56k
DAST-P	3	58k
DAST-P	4	58k
DAST-L	1	56k
DAST-L	2	56k
DAST-L	3	56k
DAST-L	4	56k
DAST-S	1	58k
DAST-S	2	59k
DAST-S	3	60k
DAST-S	4	60k
DPO	-	15k
mDPO	1	50k
mDPO	2	55k
mDPO	3	58k
mDPO	4	58k
DPO-D	1	58k
DPO-D	2	59k
DPO-D	3	60k
DPO-D	4	60k

Table 4: .

When training the models, we fix the training steps and ensure that all the models can be trained to convergences. Although the training data size of different methods are different, fixed training steps in total can maintain fairness for all the methods.

When decoding, the temperature is also set to 0.2 to be consistent with the sampling setting. All the models are quantified using float16 (fp16) to load and save parameters. The vLLM library (Kwon et al., 2023) [9] is utilized to accelerate the generation. All the experiments are conducted on 4 $\times$ NVIDIA A100-40GB GPUs.

[8] https://huggingface.co/meta-llama/Llama-3.1-8B

[9] https://github.com/vllm-project/vllm