Extended Abstract

Motivation   Recent researches in generative language models (LMs) have led to impressive results in both natural language understanding and generation. However, fine-tuning small-scale LMs to perform well across different reasoning tasks, such as instruction following and math reasoning, remains a challenge. To solve this issue, we investigate the performance of reinforcement learning (RL) based fine-tuning methods on a compact, open-source LM (Qwen2.5-0.5B Base) to understand trade-offs over preference-based alignment and domain-specific adaptation.

Method   We compared three fine-tuning techniques: supervised fine-tuning (SFT), Direct Preference Optimization (DPO), and Reinforce Leave-One-Out (RLOO). RLOO is explored by training and comparing multiple reward models: DeBERTa, DistilBERT, and Siamese DistilBERT. These are tested on the instruction following task with Smoltalk and Ultrafeedback datasets. Using additonal synthetic data and best-of-N sampling approach are extended based on the supervised fine-tuned model on math reasoning task. This is verified on the Countdown dataset.

Implementation   All models are fine-tuned using both full-parameter and Low-Rank Adaptation (LoRA) configurations. We use data augmentation via GPT-4o-generated synthetic samples and inference-time test via best-of-N sampling using external verifier functions to select responses.

Results   Experimental results show that using DPO further improve the results compared with SFT in both full and LoRA setups. Between the RLOO variants, the model using DeBERTa as a reward model generates the highest alignment score. For the math reasoning task, using synthetic data slightly improves performance, while best-of-N sampling with an external verifier generates significant gains, achieving over 0.81 accuracy, more than double that of using SFT.

Discussion   The results indicate that lightweight LMs can achieve reasonable performance after effective fine-tuning and using assistant tools. The quality of the reward model and the diversity of sampled responses are important to RLOO. In addition, using external verifiers with best-of-N sampling introduces a low-cost treatment to improve prediction accuracy during math reasoning.

Conclusion   This work demonstrates the feasibility and effectiveness of aligning small language models across diverse reasoning tasks using a combination of RL fine-tuning strategies, synthetic data, and best-of-N sampling. It highlights practical considerations for optimizing small-scale LM performance for downstream tasks under constrained computation resources.

RL Fine-Tuning of Language Model for Instruction Following and Math Reasoning

Yifu Han
Department of Energy Science & Engineering
Stanford University
yifu@stanford.edu

Geo Zhang
Department of Energy Science & Engineering
Stanford University
gmzhang@stanford.edu

Abstract

This study investigates the effectiveness of reinforcement learning (RL) fine-tuning techniques on a compact language model (Qwen2.5-0.5B Base) for two challenging tasks: instruction following and mathematical reasoning. We compare supervised fine-tuning (SFT), Direct Preference Optimization (DPO) using preference-labeled data, and Reinforce Leave-One-Out (RLOO) with reward models. Our experiments show that RLOO with DeBERTa reward modeling achieves the best alignment, while DPO provides strong and consistent results. For math reasoing tasks, synthetic data augmentation and best-of-N sampling with an external verifier significantly improve accuracy, showing the potential of combining fine-tuning with inference-time tools. This study highlights key trade-offs and practical strategies for training lightweight, task-aligned small-scale language models.

1 Introduction

Generative language models have demonstrated impressive capabilities in various natural language processing (NLP) tasks, ranging from open-domain question answering to logical inference and symbolic mathematics. However, achieving high performance across diverse domains still needs to be explored. In addition, different fine-tuning techniques including RL should be compared.

In this study, we explore the implementation of RL algorithms for fine-tuning language models (LMs) to improve their performance on two tasks, instruction following and mathematical reasoning. Instruction following (Cui et al., 2023) task is designed to test the model ability to generate high quality responses using natural language prompts, especially if the instruction is unseen in the training set and of different domain and difficulty levels. For mathematical reasoning, Countdown game is considered, which is a task to combine input numbers with arithmetic operations to get a target number (Gandhi et al., 2024). This task will evaluate the model ability to in solving math problems. We focus on training a relatively small language model, Qwen2.5-0.5B Base (Hui et al., 2024), to assess the performance on various fine-tuning techniques. Our goal is to better understand the capabilities and limitations of lightweight LMs in human-aligned learning settings.

We apply three different approaches: supervised fine-tuning (SFT), Direct Preference Optimization (DPO) (Rafailov et al., 2023), and Reinforce Leave-One-Out (RLOO) (Ahmadian et al., 2024). These methods are used to fine-tune language model on instruction-following task, where the objective is to produce responses that align well with human preference. In the case of RLOO, we evaluate different reward models, including DeBERTa (He et al., 2020), DistilBERT (Sanh et al., 2019), and Siamese DistilBERT (Reimers and Gurevych, 2019). The DistilBERT and Siamese DistilBERT reward models are trained to evaluate alignment quality, and their impact on final policy performance is compared.

To further improve the mathematical reasoning capabilities of the fine-tuned Qwen model, we construct a high-quality synthetic dataset consisting of 1600 examples based on the Countdown

dataset. This dataset considers multi-step numerical reasoning and arithmetic composition. We apply GPT-4o (Hurst et al., 2024) to assist with both problem generation and solution verification, ensuring the training data quality. In addition, we experiment with a best-of-N sampling strategy (Chow et al., 2024), guided by an external verifier, to improve the reliability and correctness of model predictions. This approach significantly improves model performance on the math reasoning task, suggesting that combining fine-tuned policies with external tools can generate more accurate results.

Overall, this work contributes practical insights into the trade-offs between different RL-based fine-tuning algorithms for language models and demonstrates effective strategies for improving performance in challenging downstream tasks such as instruction following and mathematical reasoning.

2 Related Work

Using external tools is an important approach to improve the performance of LLMs. Jin et al. (2025) proposed Search-R1, an RL framework that combines search engine interaction with proximal policy optimization (PPO) and group relative policy optimization (GRPO) training, which generates better results compared with the traditional retrieval-augmented generation methods. Gehring et al. (2024) introduced an executive feedback workflow, improving code generation capability in external test cases. Schick et al. (2023) showed that LLMs can learn to use various tools correctly through fine-tuning, learning when and how to call the appropriate APIs for each task.

The use of best-of-N sampling guided by an external verifier can be used for improving the reliability and correctness of language model outputs. This strategy involves generating multiple candidate responses from the model and selecting the best based on external evaluation criteria Chow et al. (2024). Zhou et al. (2023) proposed the least-to-most prompting and demonstrated that combining multiple outputs with verification significantly improve reasoning performance. Similarly, Madaan et al. (2023) introduced self-refine, where a verifier model iteratively improves responses by selecting the best between generated candidates. Zhang et al. (2023) further explored this direction by training verifier models to score and rerank candidate outputs in a language model cascades framework, which generates more factual and consistent generations. These methods allow for more accurate and robust outputs, which is important in mathematical reasoning and code generation. Our work applies best-of-N sampling with an external verifier to improve accuracy in arithmetic composition tasks.

3 Method

DPO is a reinforcement learning method for aligning language models with human preferences by directly optimizing a loss that favors preferred responses over less preferred ones. Given a dataset of preference pairs (x, y^+, y^-), where x is the prompt, y^+ is the preferred response, and y^- is the less preferred one, DPO minimizes a contrastive loss derived from the reward difference between responses. Assuming the reward function is the log-ratio between the fine-tuned policy π_θ and a reference policy π_{ref}, the objective becomes:

$$\mathcal{L}_{\text{DPO}}(\theta) = -\log \left[\frac{\exp\left(\beta \cdot \log \frac{\pi_\theta(y^+|x)}{\pi_{\text{ref}}(y^+|x)}\right)}{\exp\left(\beta \cdot \log \frac{\pi_\theta(y^+|x)}{\pi_{\text{ref}}(y^+|x)}\right) + \exp\left(\beta \cdot \log \frac{\pi_\theta(y^-|x)}{\pi_{\text{ref}}(y^-|x)}\right)} \right]$$

where β is a temperature parameter that controls the sharpness of preference weighting. This DPO implementation allows stable and efficient preference optimization without reward modeling.

RLOO is a policy gradient based algorithm to fine-tune language models by reducing the variance of gradient estimates using a leave-one-out baseline. Given a set of N sampled responses $\{y_1, \ldots, y_N\}$ to a prompt x, each response y_i is assigned a scalar reward r_i, computed using a separate reward model or preference function, as shown in Fig. 1. The gradient estimate for policy π_θ is computed as:

$$\nabla_\theta \mathcal{L}_{\text{RLOO}}(\theta) = -\frac{1}{N} \sum_{i=1}^{N} \left(r_i - \frac{1}{N-1} \sum_{j \neq i} r_j \right) \nabla_\theta \log \pi_\theta(y_i \mid x)$$

where $\frac{1}{N-1} \sum_{j \neq i} r_j$ is the leave-one-out baseline, effectively reducing variance in the gradient estimate. This enables stable and sample efficient fine-tuning of language models with explicit preference to the high-reward responses.
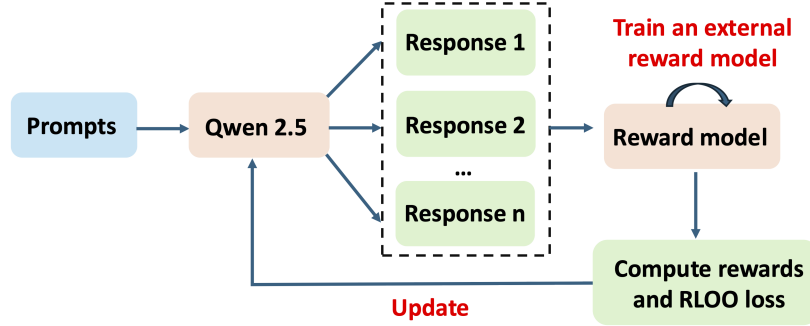

Figure 1: Overview of RLOO method with an external reward model trained and uilized for scorer.

In RLOO, a critical component is the reward model, which assigns scores to generated responses. The reward model can be based on human feedback, heuristic algorithms, or learned models. In this work, we use pretrained transformer-based models, specifically variants of BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019), which is effective in a wide range of NLP tasks. To explore the trade-offs between performance and efficiency, we applied two another BERT variants, (1) DeBERTa (Decoding-enhanced BERT with disentangled attention) (He et al., 2020), which has strong representation capabilities and improved generalization due to the enhanced attention mechanism and disentangled positional encoding; (2) DistilBERT (Sanh et al., 2019), due to its computational efficiency. These models are adapted by adding a regression head to output scalar reward scores and fine-tuned on the offline UltraFeedback dataset Cui et al. (2023). In addition to the standard single-input reward model, we also tested a Siamese BERT structure (Reimers and Gurevych, 2019) for direct preference modeling. Both the preferred and dispreferred responses (conditioned on the same prompt) are passed independently through a shared BERT encoder to produce two scalar scores and then the model is trained using the Bradley-Terry objective, which is expressed as

$$\mathcal{L}_{\text{BT}}(\phi) = \max_{\phi} \ \mathbb{E}_{(x,y^+,y^-) \sim \mathcal{D}_{\text{pref}}} \left[\log \sigma \left(r_\phi(x, y^+) - r_\phi(x, y^-) \right) \right]$$

On the math reasoning task, due to the relative small size of the dataset, we explored using synthetic data to augment the fine-tuning procedure. The synthetic data is generated through input prompts into GPT-4o (Hurst et al., 2024) as shown the process in Fig. 2.
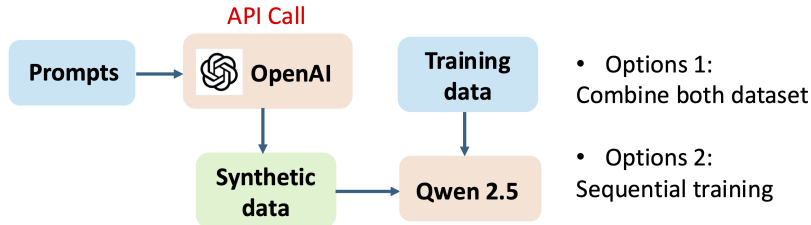

Figure 2: Generation of synthetic data using ChatGPT and utilization in model training.

To improve the capabilities, we implemented a best-of-N sampling strategy (Chow et al., 2024) and incorporated an external tool as critics to select the most appropriate response, as illustrated in Fig. 3.
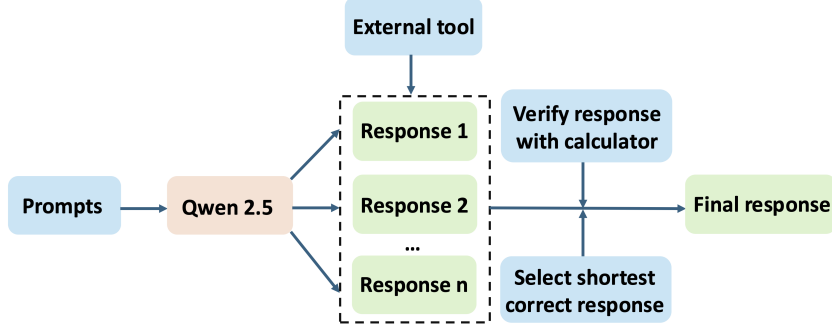
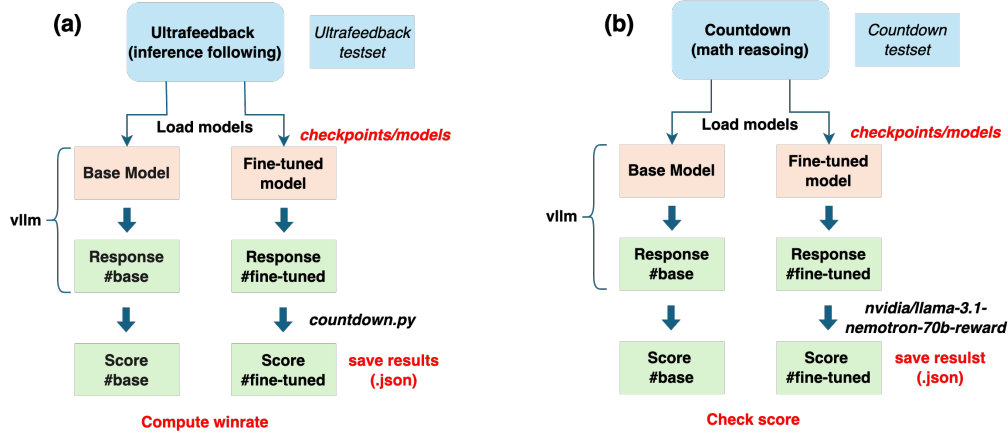Figure 3: Use best-of-N sampling strategy and external tool as critics.

Figure 4: The evaluation process of two tasks

4 Experimental Setup

We have performed two supervised fine-tuning runs for the Qwen 2.5-0.5B baseline model on a single NVIDIA A100 GPU: a LoRA run for instruction following task on the SmolTalk corpus (learning-rate 5e-6, 3 epochs, ≈11 h) and a full-parameter run for math reasoning on the Countdown Warmstart data (same learning rate, 5 epochs, ≈10 min). Evaluation procedure used vLLM for generation and was performed on 200 randomly selected SmolTalk test samples, and 1000 randomly sampled Countdown-Tasks-3to4 problems. These two cases are then scored with the Llama 3.1 Nemotron-70B reward model and an exact-match countdown.py script, respectively.

5 Results

5.1 Quantitative Evaluation

Table.1 compares the performance of different training methods including SFT, DPO, and RLOO using both full and LoRA fine-tuning setups, as well as different reward functions for RLOO. The first row shows the winning-rate evaluation metric, where DPO achieves higher winning-rate than SFT in both Full (0.605 vs. 0.495) and LoRA (0.665 vs. 0.575) settings. Between RLOO variants, the model trained with reward model of DeBERTa achieves the highest score (0.695), which is better than both SFT and DPO. The second row shows an additional evaluation based on leader-board, where SFT and DPO perform comparably well, with DPO Full reaching 0.87. RLOO with DistilBERT and Siamese DistilBERT also perform competitively (both scoring 0.8025), but achieves slightly lower winning-rate compared to DPO. Overall, the results show that RLOO with a well-designed reward function generate better results than the SFT on certain metrics, while DPO consistently generate

better results across both evaluation criteria. We also find that by using the LoRA adapters with the rank of 8,we achieved comparable performance than full parameter fine-tuning while reducing GPU memory and training time more than 50%.

Table 1: Comparison of models using different training methods and reward functions.

	SFT (Full/LoRa)	DPO (Full/LoRa)	RLOO (reward[1])	RLOO (reward[2])	RLOO (reward[3])
Score[†]	0.495 / 0.575	0.605 / 0.665	0.695	0.530	0.535
Score[‡]	0.79 / 0.885	0.87 / 0.8525	0.765	0.8025	0.8025

Table 2: Comparison of SFT (full parameters fine-tuning) with synthetic data and using external tool.

	SFT (full)	Synthetic Data (SFT full)	External Tool (SFT full)
Score*	0.37	0.3835	0.811

Table.2 presents a comparison of performance scores on the math reasoning task. The baseline model, trained with SFT data, achieves a score of 0.37. When synthetic data is incorporated into the same SFT framework, performance improves slightly to 0.3835, indicating a modest benefit from augmenting the training set. However, the most significant improvement comes from integrating an external tool at inference time, which boosts the score substantially to 0.811. This highlights the effectiveness of using an external verifier or reasoning tool to select or validate generated answers, showing that test time inference can achieve significantly improve performance and accuracy.

Fig.5 illustrates the impact of temperature and the number of samples used in best-of-N sampling on the performance in the math reasoning task, as evaluated by an external verifier. In this experiment, we evaluate three sample sizes: 3, 5, and 10. The results show that increasing the number of samples consistently improves performance over all temperature settings, with the best scores achieved when using 10 samples. As temperature increases, the scores initially improve due to greater sample diversity, but tend to decline at higher temperatures, particularly when using fewer samples. The highest score is obtained with 10 samples at a temperature of about 0.7. This suggests that using moderate temperature with a larger sample allows the external verifier to more effectively select accurate solutions during the test-time.
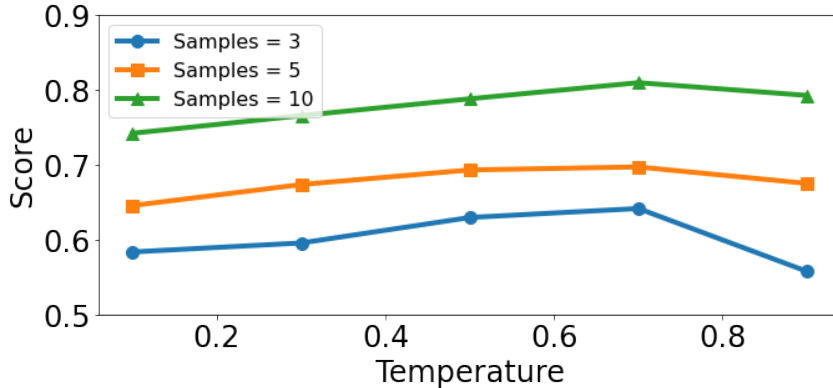

Figure 5: Scores achieved for the math reasoning task by using the external verifier for different numbers of best-of-N sampling and temperature.

5.2 Qualitative Analysis

To better understand model behavior across tasks, we conducted qualitative analysis of representative model output from each fine-tuning model.

For the instruction-following task, output from SFT model tend to be generic or overly verbose, occasionally missing the intent of the prompt. While the model fine-tuned by DPO often generates more precise and human-aligned answers, due to its preference-based optimization. For example, when asked to summarize a technical concept, DPO responses were more concise and accurate than SFT. The RLOO model, using DeBERTa reward model, generates outputs with more consistent tone and relevance, although it occasionally produces repetitive phrasing due to reward overfitting.

In the math reasoning task, the base SFT model frequently fails to complete all computation steps or makes arithmetic mistakes. The inclusion of GPT-4o-generated synthetic examples improves multi-step reasoning slightly. However, the most significant improvement arises from applying best-of-N sampling with an external verifier. For example, when asked to reach a target number using a set of integers, the verifier consistently selected correct outputs from between diverse candidates.

These results indicate that while fine-tuning improves alignment and task specificity, inference-time strategies, such as verifier-guided sampling provide a powerful, complementary mechanism for ensuring high-quality output, especially for precision-sensitive domains like mathematical reasoning.

6 Discussion

For the RLOO algorithm, two factors are essential for its performance: the number of responses generated per query and the quality of the reward function. The number of sampled responses evaluated during training directly influences the model capacity for exploration. Generating more responses per prompt allows the learning algorithm to better explore the policy space, but this comes at the cost of increased computational resources. Second, the reward function is important for guiding the model learning for RLOO. Reward models trained on high-quality, and task specific preference data can better capture differences in reasoning quality, leading to more consistent and reliable improvements for RLOO during the test-time.

The use of an external verifier for best-of-N sampling can significantly improve model performance in mathematical reasoning. A larger number of samples and a relatively higher temperature (e.g., 0.7) tend to generate more accurate responses.

7 Conclusion

This study explores the application of reinforcement learning fine-tuning techniques to a small-scale language model (Qwen2.5-0.5B Base) on two distinct tasks: instruction following and math reasoning. Through comparative analysis of SFT, DPO, and RLOO, we show that preference-based optimization methods, particularly DPO and RLOO with appropriate reward models, can significantly improve alignment with human feedback, even for lightweight language models.

Our experiments highlight that DeBERTa-based reward models are especially effective in the RLOO framework, while DPO achieves consistently strong performance with a simpler optimization setup. For the math reasoning task, we show that synthetic data generation using GPT-4o marginally improve performance, but the most substantial improvement comes from best-of-N sampling guided by external verifiers. This approach enables higher accuracy without additional training cost.

Overall, our findings suggest that small LMs can achieve competitive results when combined with effective fine-tuning algorithms, high-quality reward modeling, and inference-time verification. This provides a practical path for deploying efficient and aligned language models in low computation resource settings. Future work may explore extending this framework to additional domains such as code generation, symbolic logic, and multimodal tasks.

8 Team Contributions

- **Yifu Han:** Project conceptualization; implementation of SFT, DPO, RLOO, model evaluation pipeline, and best-of-N sampling; hyperparameter fine-tuning and analysis; draft of report; revision of report and completeness; project coordination.
- **Geo Zhang:** Project conceptualization; dataloader preparation and model prototyping; implementation of SFT, DPO, RLOO, and model evaluation pipeline; hyperparameter fine-tuning and analysis; revision of report and completeness; project coordination.

Changes from Proposal Due to time constraints, we were unable to explore the optimized fine-tuning of both tasks on a single model. However, we observed that sequentially fine-tuning a model, first on one task and then on the other, can lead to better performance compared to fine-tuning on a single task alone. The multi-objective RL optimization is also not considered.

References

Arash Ahmadian et al. 2024. Back to Basics: Revisiting REINFORCE-style Optimization for Learning from Human Feedback in LLMs. *arXiv preprint arXiv:2402.14740* (2024).

Yinlam Chow et al. 2024. Inference-Aware Fine-Tuning for Best-of-N Sampling in Large Language Models. *arXiv preprint arXiv:2412.15287* (2024).

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, et al. 2023. Ultrafeedback: Boosting language models with scaled ai feedback. *arXiv preprint arXiv:2310.01377* (2023).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 4171–4186.

Kanishk Gandhi, Denise Lee, Gabriel Grand, Muxin Liu, Winson Cheng, Archit Sharma, and Noah D Goodman. 2024. Stream of search (sos): Learning to search in language. *arXiv preprint arXiv:2404.03683* (2024).

Jonas Gehring, Kunhao Zheng, Jade Copet, Vegard Mella, Quentin Carbonneaux, Taco Cohen, and Gabriel Synnaeve. 2024. Rlef: Grounding code llms in execution feedback with reinforcement learning. *arXiv preprint arXiv:2410.02089* (2024).

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654* (2020).

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186* (2024).

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276* (2024).

Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516* (2025).

Aman Madaan, Zhecheng Lin, Shiyang An, Qian He, Xuefei Liu, Kai Yang, Isil Dillig, and Swarat Chaudhuri. 2023. Self-Refine: Iterative Refinement with Self-Feedback. *arXiv preprint arXiv:2303.17651* (2023).

Rafael Rafailov et al. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Advances in Neural Information Processing Systems*, Vol. 36. 53728–53741.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems* 36 (2023), 68539–68551.

Yuhuai Zhang, Jacob Hilton, Spyridon Papadimitriou, Yuntao Bai, Andy Jones, Jan Leike, Noam Nisan, Natalie McAleese, Amanda Askell, Geoffrey Irving, et al. 2023. Language Model Cascades. *arXiv preprint arXiv:2309.00238* (2023).

Denny Zhou, Nathanael Schärli, Lei Hou, Jason Wei, Xuezhi Zhou, Yi Wang, Swaroop Mishra, Ed Chi, and Quoc V Le. 2023. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. *arXiv preprint arXiv:2205.10625* (2023).

A   Implementation Details

The code is available at https://github.com/Yifu93/LLM-Reinforcement-Learning.