Reproduction of DeepSeek R1 Zero: A case study in Math Task

Ziqi Tang, Mingyu Chen

April 6, 2024

Abstract

This project aims to reproduce the improvement of reasoning ability in DeepSeek R1 Zero [6]. Specifically, we will fine-tune the Qwen-2.5 1.5B models using PPO and GRPO. Our main focus is on exploring the impact of hyperparameters on LLM fine-tuning performance.

Introduction

Large Language Models (LLMs) are increasingly being utilized for complex reasoning tasks, such as mathematical problem-solving [15, 21] and web navigation [8, 11]. Enhancing LLM reasoning capabilities through reinforcement learning (RL) has gained significant attention, as it enables improvement without relying on human-labeled trajectories. Recently, DeepSeek (DS), a newly introduced LLM [6], has made a significant impact on the AI community. Specifically, DS has demonstrated exceptional reasoning abilities, achieving performance comparable to larger models [16, 1] while maintaining remarkably low training costs. In this project, we attempt to reproduce the magic reasoning abilities in DS models with the RL techniques such as PPO and GRPO. Our main focus includes the following points:

1. Investigating the impact of different hyperparameters on LLM reasoning capabilities.
2. Comparing the performance of PPO and GRPO across different tasks.
3. Exploring ways to accelerate training and achieve better reasoning performance, such as encouraging LLM to do exploration.

Related Work

Reinforcement learning(RL) has emerged as a pivotal technique in the post-training of Large Language Models (LLMs) [3, 29, 17]. With the assistance of human-annotated preference data, RL significantly boosts model performance and robustness via consistent alignment

with human values. Many advanced LLMs have demonstrated the success of RL in alignment, e.g., ChatGPT [16], Claude [1], Gemini [9] and Deepseek [6]. RL in LLMs can generally be divided into two main categories: 1) reward-based methods, which are represented by PPO [20] and GRPO [21], 2) preference-based methods, which are represented by RLHF [3] and DPO [10]. For reward-based methods, they optimize a policy parameterized by a neural network based on a given reward function. Such approaches, particularly PPO, has become the dominant method for industry-level LLM post-training [16, 1, 9, 6]. For preference-based methods, they directly optimize using human preferences instead of relying on a predefined reward function. Early works in this area followed a two-stage pipeline: 1) optimizing a reward model using the preference dataset, and 2) refining the LLM policy based on the optimized reward model. Alternatively, a recent new line of research focuses on single-stage algorithms, including to SLiC [28], DPO [19], and its variants, such as IPO [2] and SPPO [25].

RL has also become crucial for advancing the reasoning abilities of LLMs, particularly as supervised fine-tuning (SFT) faces limitations in handling complex, multi-step reasoning tasks [26]. SFT initially provided substantial improvements in LLM reasoning capabilities, specializing pre-trained models for task-specific performance through annotated datasets [7, 14]. However, their effectiveness diminishes as task complexity grows, leading to challenges such as catastrophic forgetting and high computational cost. To mitigate these limitations, Chain-of-Thought (CoT) [23] was introduced, explicitly training models to generate intermediate reasoning steps, substantially enhancing structured reasoning tasks and improving transparency and interpretability of model outputs [24, 27, 13]. To stimulate the model's COT capabilities, RL methods are widely adopted. The approaches can mainly be categorized into two types: outcome-based rewards and process-based rewards. Outcome-based reward models explicitly reward the correctness of final answers, guiding the model toward accurate problem-solving results [12], which implicitly improves models' CoT abilities. Alternatively, process-based reward models encourage the generation of valid intermediate reasoning steps, ensuring that the reasoning process itself aligns with logical consistency and human interpretability [5, 22]. Such RL techniques offer greater flexibility and robustness in reasoning tasks, especially in scenarios where human alignment and context-sensitive reasoning are crucial.

Proposed Work

Following [6], we mainly focus on PPO and GRPO. Here we briefly introduce these two algorithms.

PPO Proximal Policy Optimization (PPO) [20] is a widely used reinforcement learning algorithm that improves policy optimization by enforcing constraints on policy updates. Formally, PPO optimize the policy through the following loss.

$$L^{\text{PPO}}(\theta) = \mathbb{E}_t \left[\min(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t) \right]$$

where:

1. $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the probability ratio between the new policy π_θ and the old policy $\pi_{\theta_{\text{old}}}$.
2. A_t is the advantage function, which estimates how much better an action is compared to the average action.
3. ϵ is a small hyperparameter (typically 0.1–0.2) that controls the clipping range and ensures that updates do not move too far away from the old policy, helping to stabilize training.

In PPO, the learning pipeline consists of two neural networks:

1. Actor Network: the actor network is parametrized by π_θ, which is responsible for selecting actions based on the current state.
2. Critic Network: the critic network evaluates how good a state is by estimating its state-value value function. In PPO, it is used to calculate the advantage function.

GRPO Gradient Reward Policy Optimization (GRPO) [21] is a recent advancement designed to address the limitations of PPO. Unlike PPO, which relies on the critic network to estimate the state-action value, GRPO directly modifies the policy gradient by incorporating a trajectory-based reward function. This modification improves sample efficiency and enables more stable policy learning. The loss of GRPO is denoted by

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\left[\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}\hat{A}_{i,t}, g(\epsilon,\hat{A}_{i,t})\right] - \beta D_{\text{KL}}[\pi_\theta\|\pi_{\text{ref}}] \quad (1)$$

where G represents the number of samples and t denotes the tokens index in the trajectory. $\hat{A}_{i,t}$ is the empirical advantage function based on the normalized reward, and $g(\epsilon, \hat{A}_{i,t}) = \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{i,t})$ is the clipping function as in PPO.

New Method: GRPO-Bias

In this section, we introduce a new algorithm, GRPO-Bias, inspired by recent advances in active learning for reinforcement learning. Compared to GRPO, GRPO-Bias empirically demonstrates improved exploration and generalization. The algorithm differs from GRPO by a simple one-line modification to its loss function, as detailed below

GRPO-Bias modifies the GRPO loss with a single-line change, namely

$$\mathcal{L}_{\text{GRPO-Bias}}(\theta) = \tag{2}$$

$$-\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\left[\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}(\hat{A}_{i,t}-c), g(\epsilon,(\hat{A}_{i,t}-c))\right]-\beta D_{\text{KL}}[\pi_\theta\|\pi_{\text{ref}}], \tag{3}$$

where c is a small (positive) bias. Specifically, GRPO-Bias amplify the space of negative samples while shrinking that of positive samples, resulting in a greater amount of probability mass that needs to be decreased than increased. In this regard, the surplus likelihood is naturally reallocated to other unexplored tokens. Through this mechanism, GRPO-Bias promotes more effective exploration compared to GRPO.

Theoretical intuition GRPO-Bias's loss can be approximated by

$$\mathcal{L}_{\text{GRPO-Bias}}(\theta) \approx \mathcal{L}_{\text{GRPO-Bias}}(\theta) + \frac{c}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}. \tag{4}$$

If we reduce the token-level loss to response-level loss, the loss can be considered as

$$\mathcal{L}_{\text{GRPO-Bias}}(\theta) \approx \mathcal{L}_{\text{GRPO-Bias}}(\theta) + c\sum_{i=1}^{G}\frac{\pi_\theta(y_i|x_i)}{\pi_{\theta_{\text{old}}}(y_i|x_i)}. \tag{5}$$

The gradient of GRPO-Bias is

$$\nabla\mathcal{L}_{\text{GRPO-Bias}}(\theta) \approx \mathcal{L}_{\text{GRPO}}(\theta) + c\sum_{i=1}^{G}\frac{1}{\pi_{\theta_{\text{old}}}(y_i|x_i)}\nabla\pi_\theta(y_i|x_i).$$

From the gradient, we can see that GRPO-Bias penalizes the likelihood of all previously seen samples, preventing θ from converging too quickly and thereby allowing for sufficient exploration, which helps avoid suboptimal results.

Datasets and Evaluation

We will mainly use some mathematical reasoning datasets for the training and evaluation. The datasets and benchmarks we will use mainly include

1. (GSM8K [4]) A dataset of grade school-level math word problems, designed to evaluate arithmetic and problem-solving skills in language models.
2. (CountDown) The Countdown game is a numbers puzzle where players use a set of randomly drawn numbers and basic arithmetic operations (+, -, ×, ÷) to reach or get as close as possible to a target number.

Evaluation Results

0.1 PPO results

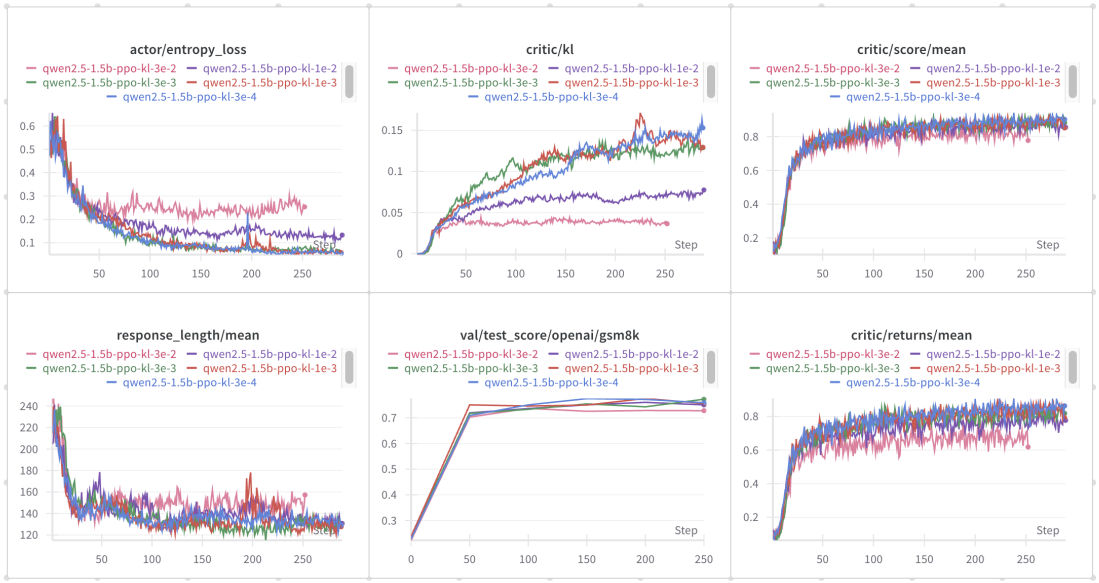

Figure 1: PPO results on GSM8K

We present our experimental results on PPO as follows. In the following experiments, we use Qwen2.5-1.5b as the base model.

GSM8K We first consider GSM8K as the dataset. The results are proposed in Fig. **??**. Specifically, we compare the performance of PPO under various KL penalty settings. As shown in the plots, most KL configurations result in stable learning behavior, with GSM8K accuracy consistently peaking between 0.75 and 0.77 across runs. This is a surprising result for a 1.5B model, especially considering that, according to the DeepSeek report [6], their Qwen2.5-1.5B model fine-tuned on r1-distilled data reaches 0.82 accuracy. This demonstrates the stability and robustness of PPO in aligning large language models through reinforcement learning.

There are some notable differences among the configurations. The entropy loss reveals that a larger KL (e.g., 3e-2) maintains higher entropy longer, suggesting more exploration, while smaller KLs promote faster convergence to lower-entropy policies. The KL divergence itself increases steadily for all runs but plateaus lower for stronger KL penalties, which is as expected. Furthermore, a milder penalty yields slightly smoother dynamics, as evidenced by more stable critic score curves. Overall, the results indicate that PPO can effectively

fine-tune Qwen2.5-1.5B for GSM8K, with mild KL settings (e.g., 1e-3 or 3e-4) offering a good balance between stability and performance.

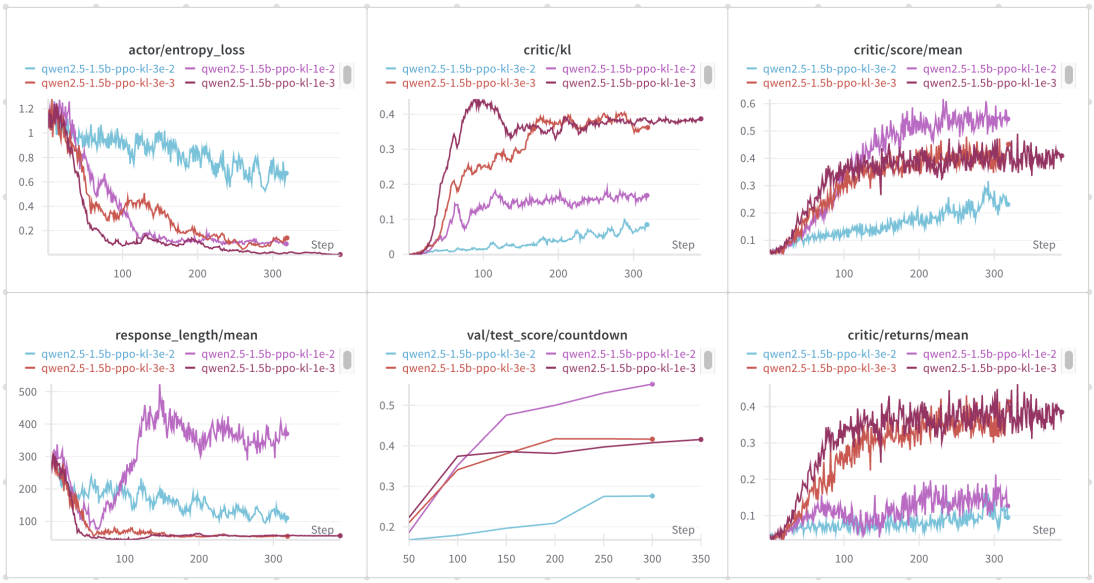

Figure 2: PPO results on CountDown

CountDown Now we move to CountDown task. The results are shown in Fig. 2. Different to the case in GSM8K, it suffices to observe that lower KL values achieve significantly better learning outcomes than the high-KL setting. The test score improves steadily for the lower KL runs, with the 1e-2 setting achieving the highest score (around 0.55), while the 3e-2 run stagnates at a much lower performance level (around 0.28). This trend is reinforced by critic score and return plots, where stronger KL penalties overly limit the policy update and impede learning. Moreover, the response length and entropy plots also show that higher KL causes the model to remain highly exploratory with long and unstable responses, while milder penalties allow more focused and efficient policy optimization. These results suggest that, on Countdown, PPO benefits from a relatively small KL constraint, allowing the model to explore meaningfully without being overly restricted, ultimately leading to better downstream performance.

0.2 GRPO and GRPO-Bias results

Due to resource limits, we cannot do hyperparameter tuning on GRPO and GRPO-Bias experiments. Alternatively, we adapt the hyperparameters reported in existing work [18].

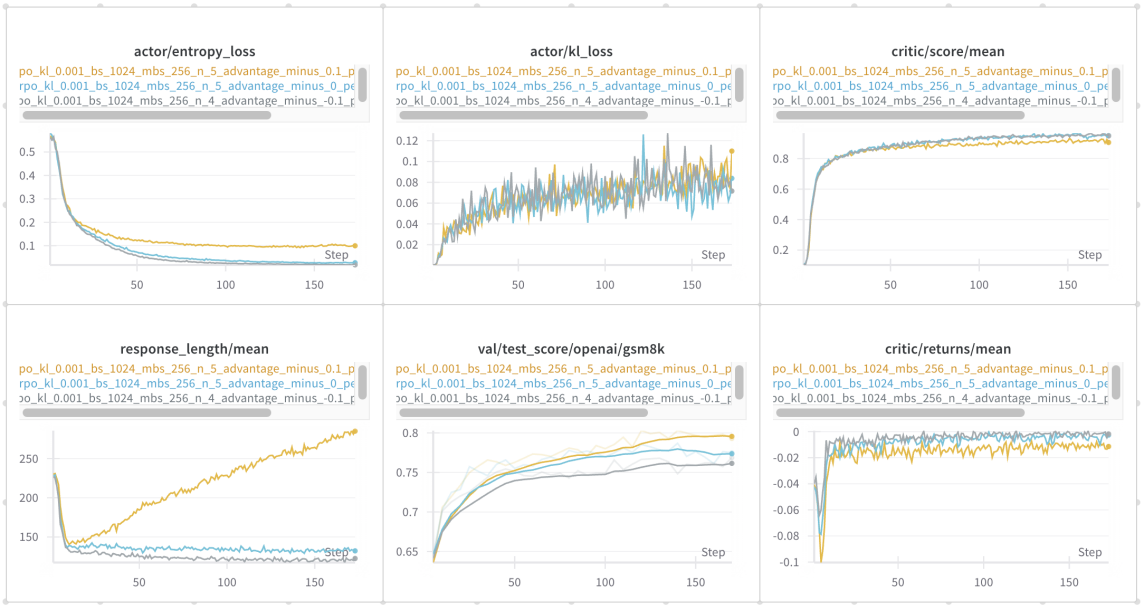

Figure 3: GRPO and GRPO-Bias results on GSM8K

GSM8K We first study GRPO and GRPO-bias on GSM8K dataset. The results are proposed in Fig. 3. To more clearly demonstrate the effect of the exploration term, we additionally construct GRPO-Bias-Inverse, whose loss is defined as follows:

$$\mathcal{L}_{\text{GRPO-Bias-Inverse}}(\theta) = \\ -\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\left[\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}(\hat{A}_{i,t}+c), g(\epsilon,(\hat{A}_{i,t}+c))\right] - \beta D_{\text{KL}}[\pi_\theta\|\pi_{\text{ref}}]. \quad (6)$$

As shown in the plots, it suffices to note that GRPO-Bias exhibits exploratory behavior, where its test score is lower than that of GRPO and GRPO-Bias-Inverse in the early steps of training, but significantly outperforms the baselines in the later steps. Moreover, the training score of GRPO-Bias is lower than that of the baselines. This is primarily because our algorithm pushes the policy away from the current policy, preventing it from converging to the optimality on the training data, but instead alleviating overfitting and achieving better generalization. More interestingly, GRPO-Bias exhibits significantly higher entropy and response length compared to the baselines. This observation further supports the exploration hypothesis: by "squeezing in" the likelihood of negative samples into the space outside the current samples, the algorithm is able to explore responses that were previously unreachable, thereby acquiring more knowledge and achieving better performance.

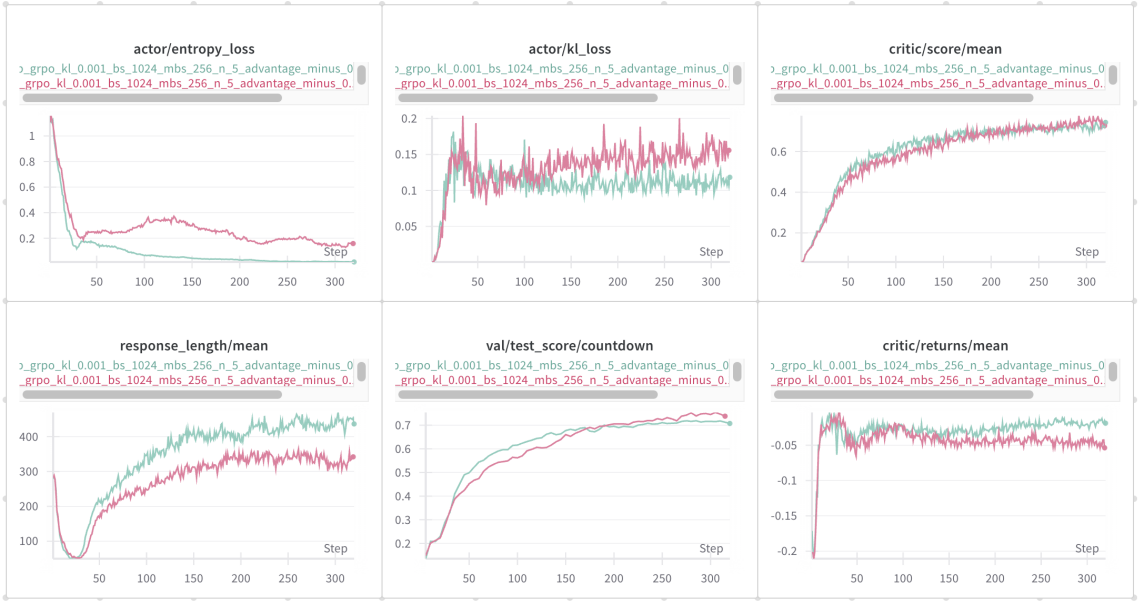

Figure 4: GRPO and GRPO-Bias results on CountDown

CountDown In the Countdown experiment, we observe similar phenomena to those in GSM8K, as GRPO-Bias demonstrates exploratory behavior and outperforms other baselines. Unlike in GSM8K, the test score in the Countdown task increases steadily without noticeable fluctuations. This is primarily due to the specific nature of the Countdown task. As shown below, the prompts in Countdown consistently follow the format:

> Using the numbers [NUM, NUM, NUM, NUM], create an equation that equals NUM. You can use basic arithmetic operations (+, -, *, /) one or multiple times, but each number can only be used once.
> Show your work in <think> </think> tags.
> Return the final equation in <answer>

In this regard, the only variation between prompts lies in the numbers NUM. This structural nature allows the LLM to generalize the information learned at each step positively across different prompts, resulting in more stable training and test performance.

Github

https://github.com/MYC000801/DS542-project/tree/final-update

Conclusion

In this work, we introduce some RL techniques, including PPO, GRPO and our specifically proposed new method GRPO-Bias, for LLM post-training to improve reasoning performance. We compare the performance of the above-mentioned algorithms on different datasets, including GSM8K and CountDown, and demonstrate significant improvements in performance of GRPO-bias compared to existing baselines. Future work includes further investigating the theoretical foundations of GRPO-Bias and evaluating its performance on larger models.

References

[1] Anthropic. Introducing claude, 2022. URL `https://www.anthropic.com/index/introducing-claude`. Accessed: 2024-12-07.

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

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

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

[5] A. Creswell, M. Shanahan, and I. Higgins. Selection-inference: Exploiting large language models for interpretable logical reasoning. *arXiv preprint arXiv:2205.09712*, 2022.

[6] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL `https://arxiv.org/abs/2501.12948`.

[7] Xiang Deng, Yu Su, Alyssa Lees, You Wu, Cong Yu, and Huan Sun. Reasonbert: Pre-trained to reason with distant supervision. *arXiv preprint arXiv:2109.04912*, 2021.

[8] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114, 2023.

[9] Google. Introducing gemini: our largest and most capable ai model, 2023. URL `https://blog.google/technology/ai/google-gemini-ai/`. Accessed: 2024-12-07.

[10] Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares,

Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, et al. Direct language model alignment from online ai feedback. *arXiv preprint arXiv:2402.04792*, 2024.

[11] Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. *arXiv preprint arXiv:2401.13649*, 2024.

[12] H. Lightman, N. Kosaraju, M. Burda, I. Mordatch, and J. Schulman. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

[13] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

[14] Elita Lobo, Chirag Agarwal, and Himabindu Lakkaraju. On the impact of fine-tuning on chain-of-thought reasoning. *arXiv preprint arXiv:2411.15382*, 2024.

[15] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*, 2023.

[16] OpenAI. Introducing chatgpt, 2022. URL `https://openai.com/index/chatgpt/`. Accessed: 2024-12-07.

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

[18] Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. Tinyzero. https://github.com/Jiayi-Pan/TinyZero, 2025. Accessed: 2025-01-24.

[19] 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, 2024.

[20] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[21] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[22] J. Uesato, S. Bhupatiraju, and R. Kumar. Solving math word problems by combining language models with symbolic solvers. *arXiv preprint arXiv:2211.14275*, 2022.

[23] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[24] Yiran Wu, Feiran Jia, Shaokun Zhang, Hangyu Li, Erkang Zhu, Yue Wang, Yin Tat Lee, Richard Peng, Qingyun Wu, and Chi Wang. An empirical study on challenging math problem solving with gpt-4. *arXiv e-prints*, pages arXiv–2306, 2023.

[25] Yue Wu, Zhiqing Sun, Huizhuo Yuan, Kaixuan Ji, Yiming Yang, and Quanquan Gu. Self-play preference optimization for language model alignment. *arXiv preprint arXiv:2405.00675*, 2024.

[26] F. Xu, Q. Hao, Z. Zong, J. Wang, Y. Zhang, J. Wang, X. Lan, J. Gong, T. Ouyang, F. Meng, C. Shao, Y. Yan, Q. Yang, Y. Song, S. Ren, X. Hu, Y. Li, J. Feng, C. Gao, and Y. Li. Towards large reasoning models: A survey of reinforced reasoning with large language models. *arXiv preprint arXiv:2501.09686*, 2025.

[27] Mengxue Zhang, Zichao Wang, Zhichao Yang, Weiqi Feng, and Andrew Lan. Interpretable math word problem solution generation via step-by-step planning. *arXiv preprint arXiv:2306.00784*, 2023.

[28] Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023.

[29] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.