Extended Abstract

Motivation Mathematical reasoning remains a significant challenge for large language models, particularly in tasks that require structured planning and symbolic manipulation. While large models excel in such domains, their computational demands limit practical deployment. In this project, we explore how to improve the performance of a small model, Qwen2.5-0.5BQwen et al. (2025), on the Countdown arithmetic game—a task that requires composing a sequence of operations to reach a target number. Our goal is to evaluate whether supervised and reinforcement learning-based fine-tuning can enhance the reasoning capabilities of compact models, offering a path toward efficient and scalable solutions in resource-constrained settings.

Method We adopt a multi-stage post-training pipeline. The process begins with supervised fine-tuning (SFT) to provide a strong initialization. This is followed by a reinforcement learning (RL) phase, where we explore three algorithms—RLOOAhmadian et al. (2024), DPORafailov et al. (2024), and XPOXie et al. (2024)—to further refine model behavior based on preference or reward signals. To promote generalization, we additionally incorporate curriculum learning by introducing more complex prompts in a second RL stage. Our investigation focuses on comparing these fine-tuning strategies within a unified framework.

Implementation We use Qwen2.5-0.5B as the base model. SFT is conducted on a warm-up dataset to help the model internalize basic reasoning and formatting patterns for Countdown. For DPO and XPO, we construct a preference dataset using the Countdown 3-to-4 dataset by sampling model completions and ranking them with a rule-based reward function. RLOO is trained online with on-policy sampling and per-batch reward normalization. In the curriculum learning phase, we continue RL fine-tuning using the Countdown 6 dataset, which includes more complex arithmetic prompts. All models are implemented in PyTorch, using HuggingFace libraries for model and data handling, and vLLM for efficient inference and sampling.

Results We evaluate performance using the official Countdown leaderboard. The base Qwen2.5-0.5B model scores 0.10, indicating limited reasoning ability out of the box. SFT improves performance to 0.3643, establishing a strong baseline. All RL methods—RLOO, DPO, and XPO—achieve scores above 0.30, comparable to SFT. The curriculum learning model, trained on Countdown 6 after RL on Countdown 3-to-4, also achieves a similar score. Across all methods, leaderboard accuracy remains within a narrow band. However, rule-based evaluation shows RL methods outperform SFT in both correctness and format.

Discussion Our results show that reinforcement learning methods—RLOO, DPO, and XPO—achieve performance comparable to supervised fine-tuning on the leaderboard, but offer measurable improvements in symbolic reasoning behavior based on rule-based metrics. While they do not lead to significant leaderboard gains, they consistently enhance format validity and arithmetic correctness relative to the SFT baseline. We also explore curriculum learning by continuing RL training on a more complex dataset (Countdown 6). Despite expectations that increased input complexity might boost generalization, performance remained similar to models trained only on simpler prompts. These findings highlight the need for deeper analysis into model behavior, reward design, and training dynamics to better leverage reinforcement and curriculum learning for symbolic reasoning in small models.

Conclusion In this project, we investigated reinforcement learning strategies to improve the performance of a small language model, Qwen2.5-0.5B, on the Countdown math reasoning task. We began with supervised fine-tuning, followed by three reinforcement learning methods—RLOO, DPO, and XPO—and extended training using curriculum learning.

All methods achieved performance on par with the SFT baseline. While these approaches are viable, they did not produce additional improvements. This motivates further work to understand their limitations and to explore more effective fine-tuning strategies for symbolic reasoning in compact models.

Mastering Mathematical Reasoning in Large Language Models via Self-Play and Reinforcement Learning

Fan Wang
Department of Computer Science
Stanford University
wang420@stanford.edu

Abstract

Mathematical reasoning in language models remains a challenging problem, especially for small models with limited capacity. In this work, we investigate how reinforcement learning techniques can enhance the performance of a compact model, Qwen2.5-0.5B, on the Countdown arithmetic game—a task requiring precise symbolic manipulation and logical planning. We follow a multi-stage training pipeline, beginning with supervised fine-tuning (SFT), followed by reinforcement learning using Reinforce Leave-One-Out (RLOO), Direct Preference Optimization (DPO), and Exploratory Preference Optimization (XPO). We also explore a curriculum learning strategy to gradually increase task complexity. Our results demonstrate that reinforcement learning can reinforce structured reasoning behaviors learned during SFT, with all methods achieving strong performance on the public leaderboard. This work highlights the potential of combining lightweight architectures with targeted training strategies for symbolic reasoning tasks.

1 Introduction

Mathematical reasoning remains a central challenge for large language models (LLMs), particularly in tasks requiring multi-step planning, symbolic manipulation, and strict constraint satisfaction. The Countdown arithmetic game exemplifies this difficulty: given a set of input numbers, the model must construct a valid sequence of operations to reach a target value. Solving such problems requires not only language fluency, but also precise logical and arithmetic reasoning.

Although recent advancements in reinforcement learning from human feedback (RLHF)Ouyang et al. (2022) have boosted LLM performance on instruction-following and general reasoning tasks, these methods have largely been developed for and evaluated on large-scale models. This raises a key question: Can small language models also benefit from reinforcement learning-based fine-tuning for complex symbolic tasks?

In this work, we address this question by targeting Qwen2.5-0.5B, a compact open-source model, and aiming to improve its performance on the Countdown task through a structured, multi-stage training pipeline. Our approach begins with supervised fine-tuning (SFT) to teach basic problem-solving strategies. We then apply three reinforcement learning methods—Reinforce Leave-One-Out (RLOO), Direct Preference Optimization (DPO), and Exploratory Preference Optimization (XPO)—to further shape the model's outputs using reward-based signals.

To promote generalization and robustness, we also incorporate curriculum learning, gradually increasing problem complexity by introducing harder input configurations. The full training pipeline is illustrated in Figure 1.

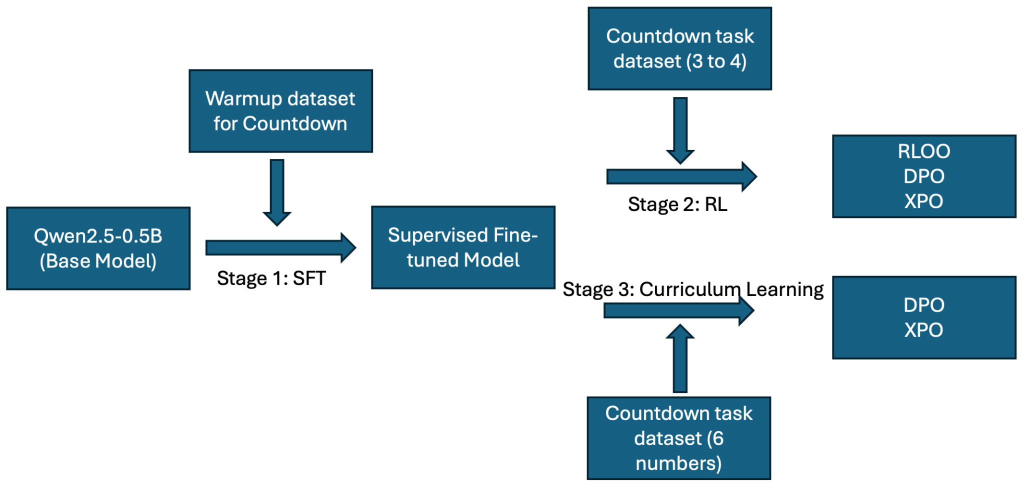
Countdown task dataset (3 to 4)
Warmup dataset for Countdown
RLOO
DPO
XPO
Stage 2: RL
Qwen2.5-0.5B (Base Model)
Supervised Fine-tuned Model
Stage 1: SFT
Stage 3: Curriculum Learning
DPO
XPO
Countdown task dataset (6 numbers)

Figure 1: Training pipeline for Countdown task, showing the progression from supervised fine-tuning to reinforcement learning and curriculum-based enhancement.

2 Related Work

Recent progress in fine-tuning large language models has largely been driven by RLHF, which enables models to align with human preferences even when explicit supervision is limited. Methods such as Proximal Policy Optimization (PPO)Schulman et al. (2017) and Direct Preference Optimization (DPO) have been successfully applied to improve instruction following and dialog generation in large-scale models like InstructGPT and ChatGPT.

However, these techniques are often tested on general language tasks and in high-capacity settings. Few studies have explored how such methods translate to smaller models or to domains that require precise symbolic reasoning. The Countdown task, which involves multi-step arithmetic planning, provides a useful test case for this purpose due to its structured and verifiable outputs.

Supervised fine-tuning (SFT) remains a strong baseline for improving model performance on structured tasks, particularly when high-quality demonstration data is available. In contrast, online RL methods such as REINFORCE and its variants (e.g., RLOO) enable policy updates based on sampled rollouts and reward feedback, at the cost of increased variance and sample inefficiency.

Exploratory preference optimization (XPO) and curriculum learning are recent extensions that aim to improve exploration and robustness. XPO encourages diversity in output trajectories during preference learning, while curriculum strategies gradually expose the model to more challenging inputs, aiding generalization.

Our work builds on these methods by applying RLOO, DPO, and XPO to a small model (Qwen2.5-0.5B) in the context of symbolic reasoning, and by systematically incorporating curriculum learning into the training process.

3 Method

Our approach follows a three-stage training pipeline designed to improve the Countdown task performance of a small language model, Qwen2.5-0.5B. These stages include supervised fine-tuning (SFT), reinforcement learning with preference optimization, and curriculum learning.

3.1 Supervised Fine-Tuning (SFT)

We begin by fine-tuning the base model on a warm-up dataset of valid problem-solution pairs from the Countdown task. The model is trained using a standard causal language modeling loss, with attention masking applied to restrict the loss to the answer portion. This teaches the model to generate valid solutions conditioned on structured inputs and initializes it for reward-driven refinement.

3.2 Reinforcement Learning Fine-Tuning

We apply three reinforcement learning algorithms to fine-tune the model beyond supervised learning: Reinforce Leave-One-Out (RLOO), Direct Preference Optimization (DPO), and Exploratory Preference Optimization (XPO). These methods leverage preference or reward signals derived from a rule-based verifier.

3.2.1 Reinforce Leave-One-Out (RLOO)

RLOO is an on-policy method based on REINFORCE that uses a leave-one-out baseline to reduce gradient variance. Given K samples of $y^{(i)}$, where $i = 1, ...k$, for a prompt x the gradient update is:

$$\nabla_\theta \mathcal{L}\text{RLOO}(\pi_\theta) = \frac{1}{k}\sum_{i=1}^{k}\left(R(y^{(i)}, x) - \frac{1}{k-1}\sum_{j=1\ j\neq i}^{k} R(y^{(j)}, x)\right)\nabla_\theta \log \pi_\theta(y^{(i)}|x) \tag{1}$$

where $R(y^{(i)}, x)$ is the reward assigned by a rule-based verifier. This baseline improves stability and sample efficiency.

3.2.2 Direct Preference Optimization (DPO)

DPO optimizes a contrastive objective using preference-labeled response pairs. Let (x, y^+, y^-) denote a prompt x, a preferred response y^+, and a dispreferred response y^-. The DPO loss is defined as:

$$\mathcal{L} \text{ of DPO}(\pi_\theta; \pi_{\text{ref}}) = \mathbb{E}(x, y^+, y^-) \sim \mathcal{D}_{\text{pref}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y^+|x)}{\pi_{\text{ref}}(y^+|x)} - \beta\log\frac{\pi_\theta(y^-|x)}{\pi_{\text{ref}}(y^-|x)}\right)\right] \tag{2}$$

where $\sigma(.)$ is the sigmoid function, and β is a temperature parameter controlling the sharpness of preference separation.

3.2.3 Exploratory Preference Optimization (XPO)

XPO extends DPO by explicitly encouraging exploration. It introduces an optimism bonus that rewards responses not well covered by the reference policy. The optimization objective for XPO at iteration t is:

$$\pi^{(t+1)} = \arg\min_{\pi\in\Pi} \alpha\sum_{i=1}^{t}\log\pi(\tau_e^{(i)}) - \sum_{(x,y^+,y^-)\in\mathcal{D}_{\text{pref}}^{(t)}}\log\sigma\left(\beta\log\frac{\pi(\tau^+)}{\pi_{\text{ref}}(\tau^+)} - \beta\log\frac{\pi(\tau^-)}{\pi_{\text{ref}}(\tau^-)}\right) \tag{3}$$

where α is the exploration coefficient and $\tau_e^{(i)}$ are responses sampled from an exploratory distribution, typically the reference or historical policy.

3.3 Curriculum Learning

To support generalization to more complex problem instances, we adopt a curriculum learning strategy. During RL training, we gradually introduce problems with a larger number of input values (e.g., 6 numbers after of 3–4), increasing symbolic complexity over time. This approach helps the model progressively build stronger reasoning capabilities while maintaining stability.

4 Experimental Setup

4.1 Tasks

Our primary task is the Countdown arithmetic game, which tests a model's ability to perform multi-step symbolic reasoning. Given a prompt consisting of a target number and a set of input numbers (typically 3–6), the model must produce a sequence of arithmetic operations (e.g., addition, subtraction, multiplication, division) that result in the target. The task requires compositional reasoning, planning, and format-adherent output generation.

We treat Countdown as both a generation and reasoning benchmark. The model must not only produce fluent outputs but also generate correct and verifiable computations. We evaluate models on both their ability to follow structural conventions and to produce accurate answers using allowed rule-based reward function.

4.2 Datasets

We use three key datasets, all drawn from or derived from Hugging Face repositories:

- Warm-up Dataset for SFT: Used to train the base model to understand the Countdown format and reasoning steps. This dataset contains prompt-solution pairs with correct outputs and structured format.
- Countdown Task Dataset (3 to 4 numbers): Used for both preference-based optimization (DPO, XPO) and online sampling (RLOO). For DPO/XPO, we construct a preference dataset by sampling multiple completions per prompt and ranking them using a rule-based verifier. For RLOO, prompts are sampled directly during training.
- Countdown Task Dataset (6 numbers): Used in the curriculum learning phase. This dataset introduces greater symbolic complexity by increasing the number of available inputs, requiring deeper reasoning chains. The 6-number prompts are introduced after initial RL convergence to improve generalization.

All datasets are preprocessed using HuggingFace tokenizers. For RLOO, prompts are dynamically sampled and scored on-the-fly. For DPO and XPO, we maintain a structured preference dataset format (x, y^+, y^-).

4.3 Metrics

We evaluate model performance using two complementary methods: the Leaderboard and the prompts from the holdout dataset of the Leaderboard.

4.3.1 Leaderboard Accuracy

Our primary evaluation metric is the leaderboard accuracy, which reflects the model's overall ability to solve Countdown prompts. This score is reported by the public leaderboard and is treated as a black-box metric. It provides a single scalar score between 0 and 1, which we use to compare the effectiveness of different training methods (SFT, RLOO, DPO, XPO).

4.3.2 Rule-Based Breakdown on Holdout Set

To gain more detailed insight into model behavior, we separately analyze a set of prompts from the leaderboard using a rule-based reward function. This function decomposes performance into:

- Correctness Score: Measures whether the model-generated expression correctly computes the target number using only the provided inputs.
- Format Score: Checks whether the output follows a valid syntactic structure (i.e., parseable, well-formed arithmetic).

All completions are generated using the vLLM inference engine under consistent decoding settings across all models.

5 Results

5.1 Quantitative Evaluation

We evaluate the performance of all models using the official Countdown leaderboard, which reports a single scalar score between 0 and 1. To supplement this black-box metric, we also evaluate models on a held-out subset of prompts using an open-source rule-based reward function that separately scores format validity and solution accuracy.

Table 1 summarizes the leaderboard scores for all models. The base Qwen2.5-0.5B model performs poorly, with a score of 0.1000, reflecting minimal out-of-the-box reasoning ability. SFT yields a substantial improvement, reaching 0.3643. All reinforcement learning methods—RLOO, DPO, and XPO—achieve scores above 0.31, with XPO scoring the highest among them at 0.3492. The curriculum learning variants, where DPO and XPO are further fine-tuned on the more complex Countdown 6 dataset, also achieve scores in a similar range.

Table 1: Leaderboard scores for all models

Model	Leaderboard Score
Qwen2.5-0.5B	0.1
SFT model	0.3643
RLOO	0.3154
DPO	0.3423
XPO	0.3492
DPO (on Countdown 6 dataset)	0.3350
XPO (on Countdown 6 dataset)	0.3430

To gain deeper insight into model behavior, we evaluate each model's predictions using a rule-based reward function on 1,000 leaderboard prompts. This function assigns two scores per example: a format score (max 0.1) for syntactic validity, and an accuracy score (max 1.0) for correctly reaching the target value using allowed numbers and operations.

Table 2 presents the aggregated results. As expected, the base model scores 0 on both metrics. The SFT model improves substantially, but is outperformed by all reinforcement learning methods in both format and accuracy. For instance, DPO achieves an accuracy score of 212 and format score of 580, compared to 146 and 388 for SFT. XPO produces the most consistently well-formed outputs, with the highest format score of 594, and a solid accuracy score of 177.

Table 2: Rule-based reward function outputs for all models on 1,000 prompts

Model	Format Score (0.1)	Accuracy Score (1.0)
Qwen2.5-0.5B	0	0
SFT model	388	146
RLOO	576	205
DPO	580	212
XPO	594	177
DPO (on Countdown 6 dataset)	481	170
XPO (on Countdown 6 dataset)	572	192

These results suggest that while leaderboard scores are relatively close across methods, reinforcement learning fine-tuning leads to measurable improvements in underlying model behavior. All RL models outperform the SFT baseline on both correctness and format metrics, with DPO and XPO achieving the highest accuracy and syntactic quality, respectively. This highlights the value of evaluating models using task-specific, interpretable metrics in addition to aggregate leaderboard scores, and supports the conclusion that RL methods are effective for improving symbolic reasoning in compact language models.

However, quantitative metrics alone do not capture how models arrive at their answers—whether they follow the correct multi-step reasoning process or simply exploit dataset artifacts. To better understand the differences in model behavior, failure modes, and response patterns, we conduct a qualitative analysis in the next section.

5.2 Qualitative Analysis

To better understand the differences between SFT and reinforcement learning (XPO), we examine specific examples where XPO successfully solved prompts that SFT failed to answer correctly. Our goal is to assess not only whether XPO produces correct outputs, but also how its arithmetic strategies differ from those of SFT.

Example 1: Direct arithmetic alignment

Input Numbers: [2315, 9984, 7509]
Target: 4790
SFT Response: `(7509 - 2315) * (9984 / 4790)`
XPO Response: `(9984 - 7509) + (2315)`
Score: SFT = 0.1, XPO = 1.0

XPO produces a clean and exact expression: $2475 + 2315 = 4790$, solving the problem perfectly. In contrast, SFT constructs a numerically complex expression involving multiplication and division, which leads to rounding errors. This case illustrates XPO's ability to discover simpler, correct expressions through exploration.

Example 2: Error reversal through reordering

Input Numbers: [867, 961, 433]
Target: 339
SFT Response: `(961 - 433) - 867`
XPO Response: `867 - 961 + 433`
Score: SFT = 0.1, XPO = 1.0

SFT's expression is valid but evaluates to the wrong value. XPO finds the correct combination and order of operations: $867 - 961 + 433 = 339$. This shows how XPO adjusts the sequence of arithmetic operations in ways that produce the correct target.

Example 3: Cleaner simplification

Input Numbers: [82, 56, 70]
Target: 96
SFT Response: `((82 + 56) - 70) + 14`
XPO Response: `70 + 82 - 56`
Score: SFT = 0.1, XPO = 1.0

SFT's solution includes an extraneous addition of 14, which is not part of the original number set. XPO, in contrast, constructs a clean, valid expression using only allowed numbers: $70 + 82 - 56 = 96$. This example highlights XPO's ability to respect input constraints while simplifying expressions.

Example 4: Deep multi-step reasoning

Input Numbers: [16, 39, 24, 53, 55]
Target: 33
SFT Response: *(missing or invalid)*
XPO Response: `(55 - 24 + 16) - 53 + 39`
Score: SFT = 0.1, XPO = 1.0

SFT fails to produce a usable output. XPO generates a longer, multi-step arithmetic chain that evaluates to the correct target. This suggests XPO's training encourages deeper composition of operations, which may help on more complex prompts.

These examples demonstrate that XPO is capable of: discovering correct number combinations that elude the SFT model; producing cleaner, constraint-compliant expressions; and adapting operation order or expression structure to better fit the target.

While overall leaderboard scores between the two models are close, these cases suggest that reinforcement learning introduces meaningful improvements in arithmetic reasoning, especially in prompts where subtle changes in operation order or numerical insight are required.

5.3 Summary of Findings

Our quantitative results show that reinforcement learning methods consistently outperform the SFT baseline in terms of rule-based correctness and format scores, even though leaderboard accuracy remains comparable. Among the RL methods, DPO and XPO demonstrate particularly strong format compliance and arithmetic accuracy across 1,000 held-out prompts.

Qualitatively, we observe that XPO generates more diverse and compositional expressions, enabling it to solve prompts that SFT fails to address. These improvements include both correct numerical reasoning and better adherence to structural constraints. In contrast, SFT tends to produce simpler but less flexible expressions, often missing the correct target or violating constraints such as one-time use of numbers.

DPO, trained with a contrastive preference loss, generates clean and well-structured outputs but with limited expression diversity. RLOO, which samples online and learns from reward differences, tends to produce deeper expressions than SFT or DPO, but does not explore as aggressively as XPO.

Together, these findings suggest that reinforcement learning introduces behavior-level improvements that may not be fully captured by scalar leaderboard metrics. Evaluating with task-specific breakdowns and qualitative analysis provides critical insight into how models reason under different training paradigms.

Table 3: Qualitative characteristics observed across fine-tuning methods

Method	Format Consistency	Arithmetic Accuracy	Expression Depth	Exploratory Behavior
SFT	Moderate	Low	Shallow	Low
RLOO	High	High	Moderate–High	Moderate
DPO	Highest	High	Moderate	Low
XPO	High	Moderate–High	High	High

6 Discussion

Our study set out to evaluate whether reinforcement learning and curriculum learning could improve the symbolic reasoning capabilities of a small language model on the Countdown task. We find that while all methods—including RLOO, DPO, XPO, and curriculum learning—are able to maintain strong performance, none substantially outperforms SFT.

One notable observation is the limited return on reward-based fine-tuning. Despite significant architectural differences among RLOO, DPO, and XPO, while leaderboard scores are similar across methods, all RL models outperform SFT in rule-based metrics. DPO achieves the highest accuracy score (212), and XPO leads in format score (594), indicating improved arithmetic precision and structural consistency through RL fine-tuning. This suggests that the knowledge acquired during supervised training is relatively stable and not easily enhanced by subsequent reward optimization—at least under the current reward design and training setup.

Similarly, our curriculum learning setup, which extends DPO fine-tuning with a more complex Countdown 6 dataset, was expected to provide additional benefit by exposing the model to harder inputs. However, performance remained unchanged. This result indicates that increasing input complexity during training does not automatically yield better generalization, and suggests that curriculum learning may require more nuanced strategies—such as dynamic difficulty scheduling or adaptive sampling—to be effective in this setting.

Overall, our findings point to several directions for future work:

- Deeper analysis of reward models and signal shaping, particularly for tasks with binary correctness like Countdown.

- Understanding training dynamics in small models, where capacity limits may constrain how much benefit can be extracted from preference-based or policy-gradient optimization.
- Combining curriculum learning with active reward modeling, rather than relying on static preference or rule-based feedback.

These questions remain open, and addressing them may unlock further gains in reasoning tasks for compact language models.

7 Conclusion

In this work, we investigated whether reinforcement learning and curriculum learning can enhance the symbolic reasoning abilities of a small language model on the Countdown arithmetic task. Using Qwen2.5-0.5B as the base model, we implemented supervised fine-tuning (SFT) followed by three reinforcement learning methods—RLOO, DPO, and XPO. We also introduced a curriculum learning variant that extends RL fine-tuning with more complex prompts from the Countdown 6 dataset.

Our findings show that all methods—reinforcement learning and curriculum learning—achieved performance on par with the SFT baseline. While these strategies are viable and maintain strong performance, leaderboard scores remained comparable, but all reinforcement learning methods produced measurable gains over SFT in rule-based accuracy and format metrics. These improvements demonstrate that RL fine-tuning enhances symbolic reasoning behaviors, even if leaderboard metrics alone may not reflect the full extent of model improvement. These results highlight the strength of SFT in this setting and suggest that further research is needed to understand how to unlock the full potential of fine-tuning for symbolic reasoning in small models.

Future work could explore improved reward shaping, adaptive task design, and training dynamics to better leverage reinforcement learning in low-capacity models and structured problem domains.

8 Team Contributions

- **Group Member 1: Fan Wang**

Changes from Proposal We proposed mult-agent RL approach in the Project Proposal, but due to resource constraint, we haven't implemented it in the final report.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. 2024. Back to Basics: Revisiting REINFORCE Style Optimization for Learning from Human Feedback in LLMs. arXiv:2402.14740 [cs.LG] `https://arxiv.org/abs/2402.14740`

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. 2022. Training language models to follow instructions with human feedback. arXiv:2203.02155 [cs.CL]

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 Technical Report. arXiv:2412.15115 [cs.CL] `https://arxiv.org/abs/2412.15115`

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. arXiv:2305.18290 [cs.LG] `https://arxiv.org/abs/2305.18290`

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG] `https://arxiv.org/abs/1707.06347`

Tengyang Xie, Dylan J. Foster, Akshay Krishnamurthy, Corby Rosset, Ahmed Awadallah, and Alexander Rakhlin. 2024. Exploratory Preference Optimization: Harnessing Implicit Q*-Approximation for Sample-Efficient RLHF. arXiv:2405.21046 [cs.LG] `https://arxiv.org/abs/2405.21046`

A Additional Experiments

We have conducted additional experiments to investigate the effects of dataset size and training duration on model performance. Specifically, we have trained RL models using different subsets of the Countdown-3to4 or 6gr dataset and different epochs. The resulting models have been evaluated using the same rule-based metrics described in the main experiments.

B Implementation Details

B.1 Datasets

We use three main datasets in our training pipeline:

- **Warm-up Dataset (for SFT)**: We fine-tune the base Qwen2.5-0.5B model using `cog_behav_all_strategies` from Hugging Face, which contains curated symbolic reasoning samples. `https://huggingface.co/datasets/Asap7772/cog_behav_all_strategies`
- **Countdown-3to4 Dataset (for RL)**: Used to generate preference datasets for DPO and XPO. We sample the first 100,000 preference pairs based on completions from the SFT model and rank them using a rule-based reward function. For RLOO, we sample the first 3,000 records. `https://huggingface.co/datasets/Jiayi-Pan/Countdown-Tasks-3to4`
- **Countdown-6gr Dataset (for Curriculum)**: Used in the second stage of curriculum training. From this dataset, we sample the first 50,000 records to generate additional preference pairs for DPO and XPO. `https://huggingface.co/datasets/alexjackson17/countdown-numbers-6-gr`

All datasets are tokenized using HuggingFace tokenizers. For DPO and XPO, preference datasets are stored as tuples of (x, y^+, y^-) with batch sampling. For RLOO, completions are sampled and scored on-the-fly using a rule-based verifier.

B.2 Preference Dataset Generation

For both the Countdown 3-to-4 and Countdown 6gr datasets, we generate 2 candidate completions per prompt using the SFT model. Sampling is performed using the `SamplingParams` configuration of vLLM:

```
SamplingParams(max_tokens=3000, temperature=0.7, top_p=0.9)
```

Responses are scored by a deterministic rule-based reward function that checks both syntactic validity and arithmetic correctness. The top- and bottom-ranked completions form (y^+, y^-) preference pairs.