

Evaluating RL efficiency improvement methods including Synthetic Data Augmentation and Any-Generation Reward Optimization for Mathematical Reasoning on Countdown Tasks

Kelvin Waititu
Stanford University
kwaititu@stanford.edu

Lin Yuan
Stanford University
liny1@stanford.edu

Juniper Jeon
Stanford University
cjeon137@stanford.edu

Extended Abstract

Large language models (LLMs) have demonstrated strong capabilities on mathematical reasoning tasks, but their performance often depends on access to large amounts of high-quality training data. While supervised fine-tuning (SFT) datasets can be expensive to curate, preference-based methods such as Identity Preference Optimization (IPO) require even more costly human annotations. Synthetic data generation offers a promising alternative by expanding training datasets without additional labeling effort. Recent advances in language models have made it possible to generate large quantities of synthetic reasoning traces, yet it remains unclear whether these synthetic examples provide meaningful benefits across different post-training paradigms. In this work, we investigate the effectiveness of synthetic data augmentation for Countdown arithmetic reasoning under both supervised and preference-based learning.

Our approach generates multiple candidate responses for each training prompt using Qwen2.5-0.5B. These responses are evaluated using task-specific scoring functions and incorporated into the training pipeline through augmentation strategies tailored to each learning objective. For SFT, high-quality synthetic responses are added to the training set as additional query-completion pairs using the shortest correct response as a selection heuristic. For IPO, generated responses are ranked according to their scores and converted into winner-loser preference pairs that are added to the original preference dataset. This framework allows us to evaluate whether synthetic supervision can improve reasoning performance without requiring additional human annotations.

We evaluate our methods on Countdown reasoning datasets containing approximately 1,200 supervised examples and over 48,000 preference pairs. Performance is measured using accuracy and pass@k metrics, which capture both single-attempt correctness and the probability of generating at least one correct solution across multiple sam-

ples. To ensure that observed improvements are not driven by randomness, each configuration is evaluated across multiple random seeds, and results are reported using mean and standard deviation statistics.

Our results reveal a clear difference between supervised and preference-based learning. Across multiple seeds, synthetic augmentation consistently improves the mean performance of SFT models while exhibiting relatively low variance, suggesting that additional reasoning demonstrations improve generalization and provide robust gains. In contrast, IPO demonstrates smaller and less consistent improvements, indicating that preference optimization is more sensitive to noise in automatically generated supervision. Although synthetic preference pairs increase the amount of available training data, inaccuracies in the generated winner-loser relationships appear to limit the quality of the resulting preference signal.

These findings suggest that the effectiveness of synthetic data augmentation depends not only on the quantity of generated data but also on how that data is incorporated into the learning objective. Methods that directly learn from demonstrations appear to benefit more readily from synthetic supervision than methods that rely on relative preference comparisons. More broadly, our results highlight the importance of evaluating both average performance and variability across seeds, as conclusions drawn from a single training run may not accurately reflect the robustness of a method.

As an exploratory extension, we investigate an off-policy reinforcement learning approach inspired by AGRO [8]. While it does not improve final reasoning accuracy, it achieves comparable performance with one-third of the training iterations, suggesting a promising direction for improving training efficiency.

Overall, our findings suggest that synthetic data augmentation is most effective in supervised settings, while preference-based methods remain more sensitive to the quality of synthetic supervision.

Abstract

Large language models have demonstrated remarkable performance on mathematical reasoning tasks, but their capabilities often depend on access to large quantities of high-quality training data. Synthetic data generation offers a scalable alternative to expensive human annotation, yet its effectiveness for improving reasoning remains poorly understood. In this work, we investigate whether synthetic data augmentation can improve performance on Countdown arithmetic reasoning tasks under Supervised Fine-Tuning (SFT) and Identity Preference Optimization (IPO). Using Qwen2.5-0.5B, we generate synthetic reasoning traces and construct augmented datasets tailored to each training paradigm. We evaluate all models using accuracy and pass@k metrics across multiple random seeds, reporting both mean and standard deviation to assess robustness.

Our results reveal a clear distinction between supervised and preference-based learning. Synthetic augmentation consistently improves SFT performance across seeds, demonstrating that additional reasoning demonstrations can enhance generalization and provide reliable gains in mathematical reasoning. In contrast, IPO exhibits smaller and less consistent improvements, suggesting that preference optimization is substantially more sensitive to noise in automatically generated supervision. As an exploratory extension, we also investigate an off-policy reinforcement learning approach inspired by AGRO and find that it achieves comparable performance with significantly fewer training iterations, highlighting a potential path toward more efficient post-training methods. Together, these findings show that the benefits of synthetic data augmentation depend strongly on the underlying learning objective and underscore the importance of supervision quality when developing data-efficient reasoning systems.

1. Introduction

Recent advances in large language models (LLMs) have led to substantial improvements across a wide range of tasks, including natural language understanding, code generation, and mathematical reasoning. In particular, reasoning-intensive benchmarks have become an important measure of model capability, as they require models to perform multi-step problem solving rather than relying solely on memorization or pattern matching. Prior work has shown that explicitly encouraging intermediate reasoning steps can significantly improve performance on arithmetic, commonsense, and symbolic reasoning tasks, highlighting the importance of reasoning-focused training and evaluation for modern language models [11].

One of the primary drivers of reasoning performance

is the quality and quantity of training data. Modern post-training techniques such as Supervised Fine-Tuning (SFT) and preference-based optimization methods rely heavily on curated datasets containing high-quality demonstrations or human preference annotations. However, collecting such data is expensive, time-consuming, and difficult to scale. This challenge has motivated growing interest in synthetic data generation, where language models themselves are used to generate additional training examples. Synthetic supervision has the potential to significantly reduce annotation costs while increasing dataset diversity, making it an attractive approach for improving model performance.

Although synthetic data generation has shown promise in a variety of domains, its effectiveness for mathematical reasoning remains unclear. In particular, different post-training paradigms may utilize synthetic supervision in fundamentally different ways. Supervised learning can directly benefit from additional reasoning demonstrations, while preference-based methods such as Identity Preference Optimization (IPO) depend on the quality of generated winner-loser relationships and preference signals [2]. As a result, synthetic data that improves one learning objective may not necessarily improve another.

In this work, we investigate the effectiveness of synthetic data augmentation on Countdown arithmetic reasoning tasks. Countdown provides a controlled reasoning environment in which models must combine arithmetic operations to reach a target number using a given set of inputs. We evaluate synthetic augmentation under two post-training paradigms: Supervised Fine-Tuning (SFT) and Identity Preference Optimization (IPO). Synthetic examples are generated using Qwen2.5-0.5B and incorporated into each training pipeline through objective-specific augmentation strategies.

Our primary research question is: **Can synthetic data augmentation improve mathematical reasoning performance on Countdown tasks, and does its effectiveness differ across supervised and preference-based learning paradigms?** We hypothesize that synthetic augmentation will improve SFT performance by increasing the diversity of available reasoning demonstrations and exposing the model to a broader range of solution strategies. We further hypothesize that IPO will benefit less from synthetic augmentation due to its greater sensitivity to noise in automatically generated preference signals. To evaluate these hypotheses, we conduct experiments across multiple random seeds and report both mean and standard deviation for accuracy and pass@k metrics.

2. Related Work

Our work builds on three main lines of prior research: reasoning-oriented language model training, synthetic data generation, and preference or reinforcement-based post-

training. Early work on chain-of-thought prompting showed that explicitly generating intermediate reasoning steps can substantially improve arithmetic, symbolic, and commonsense reasoning in large language models [11]. Follow-up work demonstrated that even simple zero-shot prompts such as “Let’s think step by step” can elicit reasoning behavior without hand-written demonstrations [5], while self-consistency decoding improved performance by sampling multiple reasoning paths and selecting the most consistent answer [9]. Other approaches, such as verifier-based reranking on GSM8K, showed that generating many candidate solutions and selecting among them can improve mathematical reasoning accuracy [4]. STaR similarly demonstrated that models can bootstrap their own reasoning ability by generating rationales, filtering for correct answers, and fine-tuning on successful reasoning traces [12]. These methods motivate our focus on generated reasoning traces, but most of them operate at inference time or rely on iterative rationale bootstrapping rather than directly comparing synthetic augmentation across SFT and preference optimization.

A second line of work studies synthetic data generation for instruction tuning and alignment. Self-Instruct showed that language models can generate their own instruction-following data, reducing dependence on human-written examples [10]. Alpaca extended this idea by fine-tuning LLaMA on 52K synthetic instruction-following demonstrations generated from a stronger teacher model [?]. WizardLM further improved synthetic instruction generation through Evol-Instruct, which automatically rewrites instructions into more complex variants [?]. Constitutional AI also uses AI-generated feedback and revisions to reduce the need for direct human supervision in alignment pipelines [3]. These works show that synthetic data can be effective, but they largely focus on general instruction following or harmlessness. In contrast, our project studies synthetic data augmentation specifically for arithmetic reasoning, where correctness is easier to verify but generated reasoning traces and preference labels may still contain noise.

Finally, our work relates to post-training methods for aligning language models with preferences and rewards. InstructGPT demonstrated the effectiveness of combining supervised fine-tuning with reinforcement learning from human feedback, but this pipeline requires expensive human preference data and reward modeling [6]. Direct Preference Optimization (DPO) simplified preference learning by directly optimizing a policy from pairwise preferences without an explicit reward model [7]. Identity Preference Optimization (IPO) further studies preference optimization under a more general theoretical framework and is one of the preference-based objectives we evaluate [?]. More recent work revisits REINFORCE-style optimization for language model alignment, showing that simpler RL algorithms can

be competitive with more complex approaches [1]. AGRO extends this direction by developing an algorithm that can use both on-policy and off-policy generations for reward optimization [8]. Our project is positioned at the intersection of these areas: we ask whether synthetic reasoning data helps equally across supervised and preference-based objectives. Our novelty lies in evaluating synthetic augmentation for Countdown reasoning under both SFT and IPO, while also exploring whether off-policy optimization can improve training efficiency.

3. Methods

Our goal is to evaluate whether synthetic data augmentation can improve mathematical reasoning performance on Countdown tasks under both supervised and preference-based learning objectives. Starting from existing Countdown datasets, we generate additional reasoning traces using a pretrained language model and incorporate these synthetic examples into both Supervised Fine-Tuning (SFT) and Identity Preference Optimization (IPO) training pipelines. Figure 1 provides an overview of the augmentation process.

3.1. Baseline Training

We first train baseline SFT and IPO models using the original Countdown datasets without augmentation. The SFT dataset consists of query-completion pairs containing arithmetic reasoning trajectories, while the IPO dataset contains winner-loser preference pairs. These baseline models serve as reference points for measuring the impact of synthetic augmentation.

3.2. Synthetic Data Generation

To generate synthetic supervision, we use Qwen2.5-0.5B to sample multiple candidate responses for each training prompt. Each candidate response is evaluated using a Countdown verifier that measures the correctness of the generated solution. The verifier provides a task-specific score that is subsequently used for filtering and ranking candidate generations.

The motivation for this approach is that arithmetic reasoning tasks admit automatic verification, allowing us to construct additional supervision without requiring human annotations. By generating multiple reasoning trajectories per prompt, we increase the diversity of available solution strategies and expose the model to a broader range of valid reasoning paths.

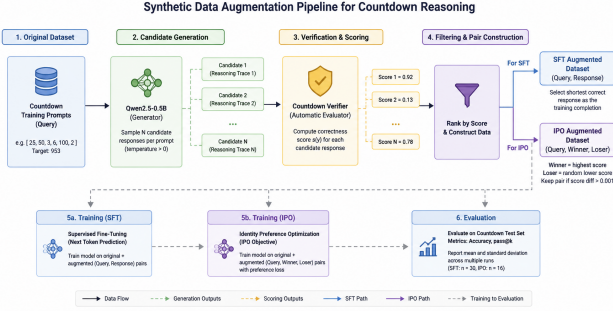


Figure 1: Synthetic data augmentation pipeline.

3.3. SFT Augmentation

For supervised fine-tuning, synthetic augmentation is performed by selecting high-quality responses from the generated candidate set. Among all correct solutions, we select the shortest response as a heuristic for the preferred reasoning trajectory. The selected synthetic responses are appended to the original training dataset as additional query-completion pairs.

This augmentation strategy assumes that shorter correct solutions provide concise reasoning signals while avoiding unnecessarily verbose trajectories. The resulting augmented dataset is then used to train a new SFT model under the same training configuration as the baseline.

3.4. IPO Augmentation

For IPO, candidate responses are first ranked according to their Countdown verifier scores. The highest-scoring response is selected as the winning response, while a randomly sampled lower-scoring response is selected as the losing response. A preference pair is retained only if the score difference exceeds a threshold of 0.001.

Formally, for a prompt x with candidate responses y_1, \dots, y_n and scoring function $s(\cdot)$, we construct a preference pair (y_w, y_l) only if

$$s(y_w) - s(y_l) > 0.001. \quad (1)$$

The resulting synthetic winner-loser pairs are added to the original IPO dataset. This filtering step reduces ambiguous preference comparisons and ensures a meaningful distinction between preferred and dispreferred outputs.

3.5. Evaluation Protocol

To account for variability arising from random initialization and training dynamics, each experiment is repeated across multiple random seeds. For SFT, we evaluate $n = 30$ independent runs, while for IPO we evaluate $n = 16$ runs. We report both the mean and standard deviation of accuracy and pass@k metrics across runs.

Algorithm 1 Synthetic Data Augmentation

- 1: **for** each training prompt x **do**
- 2: Generate candidate responses $y_{i=1}^N$ using Qwen2.5-0.5B
- 3: Score each response using the Countdown verifier
- 4: **if** SFT augmentation **then**
- 5: Select the shortest correct response y^*
- 6: Add (x, y^*) to the augmented SFT dataset
- 7: **else if** IPO augmentation **then**
- 8: Select the highest-scoring response y_w
- 9: Sample a lower-scoring response y_l
- 10: **if** $s(y_w) - s(y_l) > 0.001$ **then**
- 11: Add (x, y_w, y_l) to the augmented IPO dataset
- 12: **end if**
- 13: **end if**
- 14: **end for**

This evaluation protocol allows us to measure not only average performance but also the robustness of synthetic augmentation. Improvements that persist across multiple seeds provide stronger evidence that observed gains are attributable to the augmentation strategy rather than favorable initialization.

3.6. Novelty

Prior work has shown that synthetic reasoning traces can improve language model performance through approaches such as STaR [12] and Self-Instruct [10]. Our contribution differs in two important ways. First, we focus specifically on Countdown arithmetic reasoning, where solution correctness can be automatically verified. Second, we systematically compare the effectiveness of synthetic augmentation under both supervised (SFT) and preference-based (IPO) objectives using identical synthetic generation pipelines and multi-seed evaluation. This allows us to isolate how the choice of learning objective influences the utility of synthetic supervision.

3.7. Exploring other methods that can improve data efficiency

Besides synthetic data augmentation, we also explored an off policy RL algorithm that needs to revise RLOO method.[8]. We implemented and evaluated AGRO off-policy extension on top of a basic RLOO training pipeline, and compared the result from basic RLOO methods with synthetic data augmentation. Starting from the original on-policy RLOO loop, we added reference-regularized rewards in the update worker, so optimization is driven not only by task reward but also by a penalty on policy drift relative to a reference model. The reference model in our implementation is a past version of the network we are training. It is loaded before the training step from a saved check-

point. By doing this, it stabilizes training by preventing uncontrolled drift and reduces reward-hacking style behavior according to the paper. We also added a replay-buffer training path in the trainer, including controls for replay ratio, replay warm start, minimum buffer size, and reward-prioritized sampling.

4. Experimental Setup

4.1. Dataset

We evaluate our methods on Countdown arithmetic reasoning tasks. Each example consists of a target number and a set of input numbers, and the model must generate a valid arithmetic expression that reaches the target using the provided inputs. Countdown provides a controlled environment for evaluating multi-step mathematical reasoning because solutions can be automatically verified for correctness.

For supervised fine-tuning (SFT), we use a dataset containing approximately 1,200 query-completion pairs. For preference optimization, we use an IPO dataset containing over 48,000 winner-loser preference pairs derived from Countdown reasoning trajectories. To study the impact of synthetic augmentation, we construct augmented versions of both datasets using the procedure described in Section 3.

4.2. Models and Training

All experiments use Qwen2.5-0.5B as the base language model. We evaluate two post-training paradigms:

- **Supervised Fine-Tuning (SFT):** Models are trained on query-completion pairs using standard next-token prediction.
- **Identity Preference Optimization (IPO):** Models are trained using winner-loser preference pairs.

For each paradigm, we train both a baseline model using the original dataset and an augmented model using our synthetic data augmentation pipeline. Synthetic examples are generated using Qwen2.5-0.5B and filtered using the Countdown verifier described in Section 3.

We tuned key hyperparameters, including learning rate and batch size, over a small set of candidate values and selected the configuration that achieved the best validation performance for each training paradigm. All reported results correspond to the best-performing hyperparameter configuration.

4.3. Baselines

We compare the following methods:

- **SFT Baseline:** Trained on the original Countdown supervised dataset.

- **SFT + Synthetic Augmentation:** Trained on the original dataset augmented with synthetic query-completion pairs.
- **IPO Baseline:** Trained on the original Countdown preference dataset.
- **IPO + Synthetic Augmentation:** Trained on the original preference dataset augmented with synthetic winner-loser pairs.

These comparisons allow us to isolate the impact of synthetic data augmentation while holding the underlying training objective fixed.

4.4. Evaluation Metrics

We evaluate model performance using **Accuracy** and **Pass@k**. Accuracy measures the fraction of generated responses that correctly solve the Countdown task. Pass@k measures whether at least one correct solution is produced among k sampled generations and captures the model’s ability to generate valid reasoning trajectories under repeated sampling.

To account for variability arising from random initialization and training dynamics, each experiment is repeated across multiple random seeds. For SFT, we evaluate 30 independent runs, while for IPO we evaluate 16 runs. We report the mean and standard deviation of all evaluation metrics across runs, providing a more robust assessment of performance than a single training run.

4.5. AGRO Off-Policy Training Setup

To evaluate the effectiveness of off-policy reinforcement learning for Countdown reasoning, we adopt an aggressively off-policy training configuration inspired by AGRO. Specifically, we set the replay buffer ratio to 0.9, meaning that 90% of training samples are drawn from the replay buffer while only 10% are collected online during the current training iteration.

In addition to increasing replay utilization, we bias replay sampling toward high-quality trajectories. Rather than sampling uniformly from the replay buffer, batches with higher reward scores are assigned a higher probability of being selected for training. This prioritization strategy encourages the model to learn more frequently from successful reasoning trajectories while still maintaining some diversity through online exploration.

Our motivation for this configuration is to maximize sample efficiency and reduce reliance on expensive online rollouts. By heavily leveraging previously collected high-reward trajectories, we aim to determine whether comparable reasoning performance can be achieved with substantially fewer training iterations than standard on-policy approaches.

Table 1: Pass@ k metric across different values of k .

k	SFT	IPO	SFT Mean	SFT Std	IPO Mean	IPO Std
1	0.3513	0.3775	0.4061	0.1140	0.0002	0.0004
2	0.5112	0.5378	0.5571	0.1523	0.0003	0.0009
4	0.6347	0.6711	0.6696	0.1750	0.0006	0.0017
8	0.6976	0.7535	0.7117	0.1754	0.0013	0.0034
16	0.7400	0.8000	0.7770	0.1898	0.0025	0.0068

5. Results

5.1. Synthetic Data Augmentation

For both SFT IPO, we ran multiple iterations and found the mean and standard deviation of their respective performances. For SFT, we used $n = 30$, while for IPO, we used $n = 16$.

We included one of the training curves for each of SFT and IPO in Figure 2.

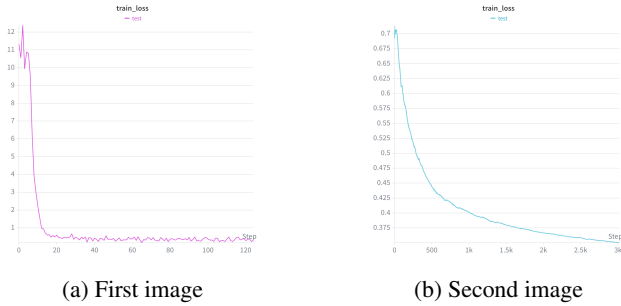


Figure 2: Training curves for augmented SFT and IPO

Table 1 shows the mean and standard deviation of the non-augmented SFT and IPO, as well as the mean and standard deviation across 30 SFT runs and 16 IPO runs. We have plotted the results across different pass@ k in Figure 2.

We noticed noticeable improvements in SFT performance with the augmented data. However, we the standard deviation is quite high and only increases as k increases. This indicates noisy improvements, which may be mitigated by taking the mean across different runs or by using a better heuristic to assess synthetic data quality.

The results for IPO were rather stark in comparison. Training loss was similar to, if slightly worse, than the non-augmented IPO curve, but when we evaluated it through pass@ k , we saw significantly worse performance. interestingly, when we opened the sample rollouts, we saw fairly reasonable outputs interspersed with long chains of repeated words, which did not occur in SFT. Further analysis is required to understand the reason for these

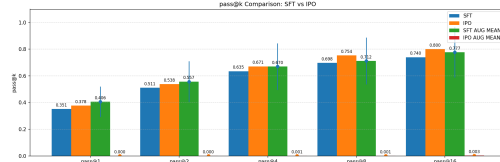


Figure 3: Comparison of pass@ k with and without synthetic data augmentation.

outputs.

Example SFT Rollout:

<think>

I will try to find a combination of operations that will give me 98. Since 98 is an even number, I will use division to make it smaller. I can divide 44 by 2 to get 22, then divide 22 by 3 to get 7.777... which is close to 8. I can use subtraction to get 19 and 35. I can subtract 19 from 35 to get 16. Then I can add 16 to 44 to get 60, which is too big. I can try adding 35 to 19 to get 54, and then adding 16 to get 60 again. So, my answer is 60.

<think>

<answer> (60 + 35) - (19 + 16)

<answer>

Example IPO Rollout:

["Step 1: We have four numbers: 44, 19, 35, and we want to create an equation that equals 98.

Step 2: We can use basic arithmetic operations (+, -, *,) and each number can only be used once.

Step 3: To make the equation easier to work with, we can start by adding the smallest numbers first. 44 + 19 = 63.

Step 4: Now we have 63 + 35 = 98.

Step 5: So, the equation is 44 + 19 +

35 = 98.
 Step 6: The final answer is 98.
 moire
 moire
 moire
 moire
 ...

5.2. AGRO Off policy training

With the training parameters mentioned in the above sections, we can get the below training results and top k performance from figure 3 and figure 4.

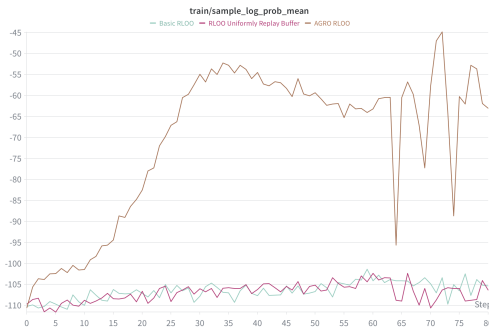


Figure 4: Average log-prob recorded at sampling time.

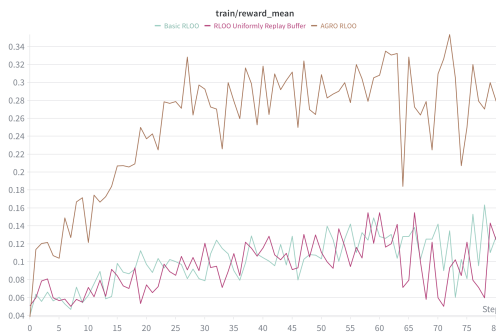


Figure 5: Mean reward during training.

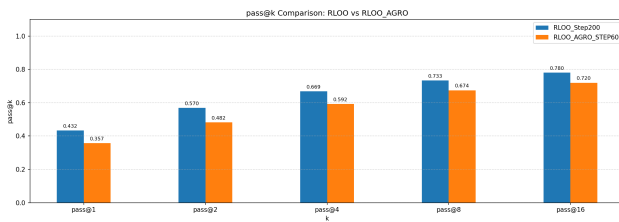


Figure 8: Comparison of pass@k with and without AGRO

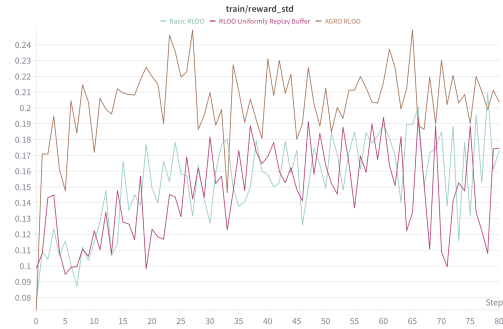


Figure 6: Reward std in training batch. Replay ratio = 0.9, reward-prioritized sampling.

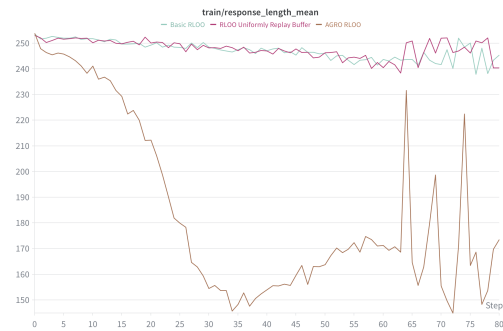


Figure 7: Response length during AGRO off-policy training.

Table 2: AGRO reaches peak reward at step 35 and comparable final pass@k at step 60, vs. 200 steps for RLOO — a **3.3× reduction** in training cost.

	RLOO	AGRO (ours)
Training steps	200	60
Peak reward step(Train)	—	35
Steps saved	—	140 (70%)
Relative speedup	1.0×	5.7×
Pass@1	0.4325	0.3575
Pass@2	0.5695	0.4823
Pass@4	0.6686	0.5919
Pass@8	0.7335	0.6745
Pass@16	0.7800	0.7200

6. Discussion

Our results highlight both the promise and limitations of synthetic data augmentation for mathematical reasoning. While synthetic augmentation consistently improved SFT performance, the gains were accompanied by relatively high variance across random seeds, suggesting that the quality of generated supervision remains an important factor. In con-

trast, IPO showed little to no improvement from synthetic preference pairs and occasionally produced degenerate outputs characterized by repetitive token generation. These findings suggest that preference-based methods are significantly more sensitive to noise in automatically generated supervision than standard supervised learning. A key challenge throughout this project was designing heuristics for selecting and ranking synthetic examples. Although arithmetic reasoning tasks benefit from automatic verification, verifier scores do not always capture the full quality of a reasoning trajectory, which can lead to noisy training signals.

Beyond Countdown reasoning, our findings have broader implications for the use of synthetic data in large language model training. Synthetic supervision offers a scalable alternative to expensive human annotation and may become increasingly important as models continue to require larger training datasets. However, our results suggest that simply increasing the amount of synthetic data is insufficient; the quality of generated examples and the learning objective used to consume them are equally important. Future work should explore stronger verifier models, improved preference-pair construction methods, and evaluation on more diverse reasoning benchmarks to better understand when synthetic data augmentation provides reliable benefits.

7. Conclusion and Future Works

Our results show that synthetic data augmentation improves performance for SFT. Generating additional candidate responses and using length as a heuristic proved effective in improving performance for SFT. However, the standard deviation is rather high; it would be worthwhile to explore alternative heuristics besides text length or employing scoring methods to improve synthetic data quality.

We unfortunately did not see similar gains with IPO. In particular, the poor performance under the pass@k metric stood out compared to its performance in training time. We did notice abnormal behavior in the rollouts, with multiple rollouts displaying long chains of single-word repeats, which was not present in SFT. We believe exploring alternative scoring methods and methods of choosing winner and loser responses to be areas of interest for future works. For improving the data efficiency, we successfully see a greatly increase at training converge speed. However we didn't improve the performance for RLOO with off policy training.

8. Team Contribution

Kelvin implemented the RLOO baseline, did extensive literature review and contributed greatly to the poster and paper.

Lin implemented the SFT baseline, worked on data efficiency improvement extension (AGRO off policy RL) and contributed to the poster and paper.

Juniper worked on the synthetic data augmentation extension and contributed to the poster and paper.

References

- [1] A. Ahmadian, C. Cremer, M. Gall'e, et al. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in language models. In *ICML*, 2024. 3
- [2] M. G. Azar et al. A general theoretical paradigm to understand learning from human preferences. *arXiv preprint arXiv:2310.12036*, 2023. 2
- [3] Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022. 3
- [4] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, J. Tworek, J. Hilton, R. Nakano, and C. Hesse. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. 3
- [5] T. Kojima, S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa. Large language models are zero-shot reasoners. In *NeurIPS*, 2022. 3
- [6] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, et al. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022. 3
- [7] R. Rafailov, A. Sharma, E. Mitchell, and C. D. Manning. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023. 3
- [8] Y. Tang, T. Cohen, D. W. Zhang, M. Valko, and R. Munos. RL-finetuning llms from on- and off-policy data with a single algorithm, 2025. 1, 3, 4
- [9] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, and D. Zhou. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2023. 3
- [10] Y. Wang, Y. Kordi, S. Mishra, A. Smith, D. Khashabi, and H. Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *ACL*, 2023. 3, 4
- [11] J. Wei, X. Wang, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022. 2, 3
- [12] E. Zelikman, Y. Wu, J. Mu, and N. Goodman. Star: Self-taught reasoner bootstrapping reasoning with reasoning. *NeurIPS*, 2022. 3, 4