Extended Abstract

Motivation Small language models (0.5B parameters) offer computational efficiency but struggle with complex reasoning tasks. We investigate curriculum learning for the countdown numbers game, where models use arithmetic operations to reach target numbers from given operands. Existing curriculum approaches use static difficulty scheduling that can cause training instability. This work examines whether adaptive curriculum scheduling based on performance feedback can improve training efficiency while avoiding model collapse in sparse reward reinforcement learning environments.

Method After training a model to output parseable results using SFT with a set of solved example problems, we develop an adaptive curriculum framework for RLOO (Reinforcement Learning from Online Optimization) training that targets 50% success rates across completions. This target maximizes the variance of advantage estimates $(R(\tau) - b)$ in the REINFORCE objective, ensuring meaningful gradient signals. We extract difficulty features from countdown problems including parse tree depth, operator diversity, and maximum intermediate values. A linear regression model predicts problem difficulty using performance feedback collected during training. The system dynamically filters problems from a fixed dataset to maintain optimal difficulty levels, preventing scenarios where all RLOO completions uniformly succeed or fail.

Implementation Our implementation uses a Qwen 0.5B model fine-tuned using SFT on solved countdown problems, followed by RLOO training with 16 completion candidates. We generate a base dataset of countdown problems with 3-4 operands covering different structural patterns. The adaptive curriculum manager monitors success rates every 50 training steps, updates the difficulty prediction model, and selects problem subsets predicted to achieve the target 50% success rate.

Results We compare three curriculum strategies over 500 training steps. No-curriculum training achieves 59.0% success rate on evaluation problems. Fixed curriculum obtained better results after hyperparameter tweaking found the goldilocks zone of problems that were neither too easy nor too hard. Our adaptive curriculum achieves 64.5% success rate while maintaining stability. However, this represents only modest improvement over the no-curriculum baseline, suggesting that curriculum benefits may be more limited than initially expected for this task and model size. Additionally, an adaptive curriculum based on difficulty estimation of existing dataset problems was implemented and shows promise, but still needs further investigation.

Discussion Our results demonstrate both opportunities and limitations of adaptive curriculum learning. While the 50% success rate target successfully prevents the catastrophic collapse observed in fixed curriculum approaches, the overall improvement over random sampling is substantial (64.5% vs 59.0%). The fixed curriculum failure illustrated the brittleness of static scheduling - rapid difficulty progression causes all RLOO completions to fail, reducing optimization to KL penalty minimization. The adaptive approach's stability suggests that performance feedback can guide curriculum scheduling, though the limited gains raise questions about curriculum learning's effectiveness for small models on this reasoning task.

Conclusion This work investigates adaptive curriculum learning for small language models in sparse reward environments. Our key finding is that while adaptive scheduling prevents catastrophic training failures seen in fixed curricula, the performance gains over random sampling are modest. The 50% success rate target maintains training stability by ensuring gradient signal variance, but curriculum learning may offer diminishing returns for small models on certain reasoning tasks. Future work should explore whether curriculum benefits scale with model size or emerge more clearly in other reasoning domains.

Project TANTALUS

Nathaniel Voorhies
Department of Computer Science
Stanford University
ncv@stanford.edu

Abstract

Small language models offer computational efficiency but struggle with complex reasoning tasks requiring sequential decision-making. We investigate adaptive curriculum learning for the countdown numbers game using RLOO (Reinforcement Learning from Online Optimization) training with sparse binary rewards. Our approach dynamically adjusts problem difficulty based on model performance feedback, targeting a 50% success rate across RLOO completions to maximize gradient signal variance. We compare four strategies: no curriculum (random sampling), fixed curriculum (predetermined progression), adaptive curriculum (performance-driven selection), and sampling adaptive curriculum (brute-force solution difficulty estimation). Results show that fixed curriculum achieves 63.4% success but can suffer instability when difficulty progression outpaces model capabilities. Our adaptive approaches achieve 64.5% (adaptive) and 65.4% (sampling adaptive) success while maintaining training stability, representing modest but consistent improvements over random sampling (59.0%). The key finding is that adaptive scheduling eliminates the need for manual hyperparameter tuning to find optimal difficulty progression, providing a robust alternative to fixed curriculum approaches. This work demonstrates both the potential and limitations of curriculum learning for small models, showing that adaptive methods prevent training instabilities while providing moderate performance gains.

1 Introduction

Mathematical reasoning represents a particularly challenging domain for small models (those with 1 billion parameters or less), as it demands precise operation sequencing, accurate intermediate calculations, and the ability to backtrack when approaches prove unfruitful. The countdown numbers game exemplifies these challenges: given a set of numbers and a target value, models must combine arithmetic operations to reach the target while using each number at most once. This task requires models to explore combinatorial search spaces, evaluate intermediate results, and select among multiple valid solution paths. These are capabilities that emerge slowly in small models through standard training approaches.

Curriculum learning, which presents training examples in order of increasing difficulty, has shown promise in improving learning efficiency in various domains. The core hypothesis is that models can build foundational skills on simpler examples before tackling more complex problems, leading to better final performance and more stable training dynamics. However, existing curriculum learning approaches for language models typically employ static difficulty schedules or simple heuristics for complexity estimation, which may not adapt to the evolving capabilities of the model during training.

Policy gradient methods their variants like RLOO or GRPO DeepSeek-AI et al. (2025) have become standard approaches for aligning language models with desired behaviors. However, these methods face significant challenges when applied to mathematical reasoning tasks with sparse binary rewards (correct/incorrect solutions). The sparsity of meaningful feedback signals can lead to training

instability, particularly when problem difficulty is not carefully calibrated to model capabilities. If problems are too easy, all completions succeed and provide no learning signal; if too hard, all completions fail and no learning signal is retrieved from the model results.

This work investigates adaptive curriculum learning for small language models in sparse reward environments, using the countdown numbers game as a testbed. We focus on a critical but underexplored question: how should curriculum difficulty progress be calibrated to maintain optimal learning signals throughout training? Our approach continuously monitors model performance to adjust problem difficulty, targeting a 50% success rate across RLOO completions to maximize the variance of advantage estimates and ensure meaningful gradient signals.

We make several contributions to understanding curriculum learning in sparse reward reinforcement learning settings. First, we demonstrate that fixed curriculum schedules, while initially promising, can lead to catastrophic training collapse when difficulty progression outpaces model learning. This failure mode occurs when all RLOO completions begin failing uniformly, eliminating the reward variance necessary for policy gradient updates. Second, we develop an adaptive curriculum framework that uses performance feedback to maintain optimal challenge levels, preventing both reward saturation (uniform success) and reward starvation (uniform failure). Third, we provide empirical analysis showing that while adaptive curriculum prevents catastrophic failures seen in fixed approaches, the overall performance gains over random sampling are modest for small models on this reasoning task.

Our experimental results reveal both the potential and limitations of curriculum learning for small language models. Comparing no curriculum (59.0% success), fixed curriculum (63.4% success), and adaptive curriculum (64.5% success), we find that curriculum timing is as critical as difficulty ordering. The adaptive approach successfully maintains training stability and provides meaningful but moderate improvements, suggesting that curriculum benefits may be more limited than initially expected for certain model sizes and reasoning tasks. These findings have implications for the broader application of curriculum learning in resource-constrained reinforcement learning scenarios where training efficiency directly impacts practical deployment feasibility.

2 Related Work

Curriculum learning, initially formalized in Bengio et al. (2009), involves training models on examples of increasing difficulty to improve learning outcomes. Recent advancements have shown particular promise for enhancing small language models' reasoning capabilities without requiring the enormous computational resources of their larger counterparts.

The DeepSeek-R1 paper demonstrates effective curriculum learning through a structured progression from simpler to complex reasoning capabilities DeepSeek-AI et al. (2025). Their multi-stage pipeline (pure RL, cold-start data incorporation, targeted RL, and comprehensive training) systematically builds reasoning skills in a difficulty-progressive sequence.

Building on this foundation, Gandhi et al. (2025) explored the causal relationship between cognitive behaviors and self-improvement capacity. Their research revealed that models exhibiting specific reasoning patterns (particularly verification and backtracking) demonstrated significantly better improvement through reinforcement learning compared to models lacking these behaviors. This explains why some language models effectively utilize additional computation while others plateau. There process by which these abilities are gained, and gained efficiently is not discussed.

For small code models specifically, recent work has demonstrated that well-designed curriculum learning strategies can substantially improve performance on complex execution tasks while having less impact on simpler completion tasks Naïr et al. (2024). This task-dependent benefit suggests that curriculum approaches are particularly valuable for complex reasoning challenges that require sequential problem-solving. However, the curriculum was divided into 3 levels based on a scalar metric of code complexity.

The R^3 (Reverse Curriculum Reinforcement Learning) technique offers another promising approach, progressively working backward from solution demonstrations to create a step-wise curriculum that enables more efficient learning Xi et al. (2024). This is similar to the process we propose, although working with fully-reasoned correct examples rather than building up skills from smaller problems.

The success of these approaches suggests that combining curriculum learning with reinforcement learning offers a promising path forward for enhancing small language models' reasoning capabilities, potentially narrowing the gap between small, efficient models and their resource-intensive counterparts.

3 Method

Our approach addresses the challenge of maintaining optimal learning signals in sparse reward reinforcement learning through adaptive curriculum scheduling. We develop a framework that continuously monitors model performance to adjust problem difficulty, preventing both the reward saturation that occurs when problems are too easy and the reward starvation that leads to training collapse when problems are too hard.

3.1 Problem Setup

Our approach addresses the challenge of maintaining optimal learning signals in sparse reward reinforcement learning through adaptive curriculum scheduling. We use the countdown numbers game as our testbed: given operands $O = \{o_1, o_2, \ldots, o_n\}$ and target value t, the model must generate an arithmetic expression using $\{+, -, \times, \div\}$ such that the expression evaluates to t and each operand is used at most once. For example, given operands $\{2, 4, 6, 8\}$ and target 20, a valid solution is $(\frac{8}{2}) \times 6 - 4 = 20$.

We frame this as a sequence generation task with binary rewards: $R = 1$ for correct solutions, $R = 0$ otherwise. We employ RLOO (REINFORCE Leave One Out) training, which optimizes the policy using:

$$L(\theta) = -\mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} \log \pi_\theta(a_t|s_t) \cdot (R(\tau) - b) \right] \tag{1}$$

where $b = \frac{1}{k} \sum_{i=1}^{k} R(\tau_i)$ is the baseline estimated from k completions.

The critical insight is that meaningful learning occurs only when there is variance in advantage estimates $(R(\tau) - b)$ across completions. When all completions receive identical rewards—either uniform success or failure—the advantage becomes zero and training reduces to KL regularization rather than task improvement. Our adaptive curriculum maintains $\sim$50% success rates to maximize this variance and ensure stable gradient signals.

3.2 Difficulty Feature Extraction

To enable adaptive curriculum scheduling, we extract comprehensive difficulty features from each countdown problem. Unlike previous approaches that rely on simple scalar metrics, we develop a multi-dimensional feature representation capturing various aspects of problem complexity:

Structural Features: We analyze the parse tree structure of potential solutions, including tree depth (number of operation levels), operator diversity (variety of arithmetic operations required), and branching patterns that indicate computational complexity.

Numerical Features: We compute target-relative metrics such as the ratio of target to maximum operand, target to operand sum, and target to operand mean. These features capture whether the target requires multiplication (target larger than sum), division (target smaller than minimum operand), or balanced operations.

Combinatorial Features: We estimate the size of the solution space through metrics like the maximum intermediate value in the parse tree.

The complete feature vector $\mathbf{f} \in \mathbb{R}^d$ for a problem with operands O, target t, intermediate values I is:

$$\mathbf{f}(O, t) = [|O|, \sum O, \max O, |\{O\}|, \max I, t, \text{tree complexity}, \text{operation complexity}] \tag{2}$$

Where "tree complexity" is a value representing the shape of the solution parse tree, and "operation complexity" is the dot product of the counts of operations and weights for operation difficulty.

3.3 Adaptive Difficulty Prediction

Our adaptive curriculum system learns to predict problem difficulty from observed model performance. We maintain a dataset $\mathcal{D} = \{(\mathbf{f}_i, s_i)\}$ where $\mathbf{f}_i$ are the extracted features for problem i and s_i is the observed success rate across k RLOO completions for that problem.

We train a regression model $g_\phi : \mathbb{R}^d \to [0, 1]$ to predict success rates:

$$\hat{s} = g_\phi(\mathbf{f}) \tag{3}$$

The model g_ϕ is updated continuously 3 as new performance data becomes available. We experiment with both linear regression and neural network architectures, with the neural network using one-hot encodings of operands, targets, and structural patterns to capture complex feature interactions.

The target success rate is set to 50%, which maximizes the entropy of binary outcomes and provides optimal variance in advantage estimates. From an information-theoretic perspective, this target ensures maximum information content per RLOO completion while maintaining meaningful gradient signals.

3.4 Curriculum Scheduling

Our adaptive curriculum operates through the following procedure:

Initialization: We begin with a diverse set of problems spanning various difficulty levels, allowing the system to collect initial performance data across the problem space.

Performance Monitoring: After every training step, we record the problem features and observed success rate across k RLOO completions. This data is added to our training dataset $\mathcal{D}$.

Model Updates: Every N steps (typically 50-200), we retrain the difficulty prediction model g_ϕ using the accumulated performance data. This allows the system to adapt to the model's evolving capabilities.

Problem Selection: For the next training phase, we filter the available problem set to select instances where $|g_\phi(\mathbf{f}) - 0.5| \leq \epsilon$ for some tolerance ϵ. This ensures we present problems predicted to achieve approximately 50% success rate.

The adaptive nature of this approach allows the curriculum to automatically adjust to the model's learning progress, maintaining optimal challenge levels throughout training without requiring manual tuning of difficulty schedules.

3.5 Baseline Comparisons

We compare our adaptive curriculum against two baseline approaches:

No Curriculum: Random sampling from the full problem distribution, representing standard training without curriculum learning.

Fixed Curriculum: A predetermined sequence of 10 difficulty levels based on initial feature-based difficulty estimates, with progression occurring at fixed intervals regardless of model performance.

This experimental design allows us to isolate the effects of adaptive scheduling versus the curriculum learning principle itself, providing insights into when and why curriculum approaches succeed or fail in sparse reward environments.

3.6 Sampling Adaptive Curriculum

In addition to the performance-feedback adaptive curriculum, we implement a second adaptive approach that estimates difficulty through brute-force solution analysis. This sampling adaptive method takes a set of problems, solves them exhaustively to find all valid solutions, and then applies the same linear regression framework to predict difficulty based on solution characteristics rather than observed model performance.

This approach mitigates the risk of transfer-learning issues if we have a representative set of problems, where the purely synthesized problems might target modes of difficulty that aren't realized in the

actual ground truth problem set, such as large operands or high amounts of division or equation complexity.

4 Experimental Setup

4.1 Model and Training Configuration

We conduct our experiments using a Qwen 0.5B language model, which provides a representative example of small, computationally efficient models suitable for resource-constrained deployment. The model is first fine-tuned using supervised learning on a dataset of countdown problems with known solutions to establish basic formatting capabilities and understanding of the task structure. This supervised fine-tuning (SFT) phase ensures that the model can generate responses in the expected format before reinforcement learning begins.

Following SFT, we apply RLOO training using identical hyperparameters across all curriculum conditions to ensure fair comparison. We use 16 completion candidates per problem ($k = 16$) to provide robust baseline estimation and sufficient statistical power for measuring success rates. The learning rate is set to 1×10^{-5} with a batch size of 1, and training proceeds for 1000 steps. All models use the same reference policy (the SFT model) for KL regularization to maintain consistent training dynamics.

4.2 Problem Dataset and Curriculum Conditions

Our synthetic problem dataset consists of countdown problems with 3-4 operands drawn from low valued integers, with targets ranging from 1-99. We generate a comprehensive set covering various structural patterns and difficulty levels, ensuring adequate representation across the problem space. For evaluation, we maintain a held-out set of 200 problems from the `Asap7772/cog_behav_all_strategies`Singh (2025) dataset that is never used during training or curriculum selection, ensuring our results measure generalization rather than memorization.

We compare four curriculum scheduling approaches:

Random Curriculum: Problems are sampled uniformly at random from the full problem distribution throughout training. This represents standard reinforcement learning without curriculum learning and serves as our primary baseline.

Fixed Curriculum: Problems are organized into 10 predetermined difficulty levels based on initial feature-based estimates. The curriculum progresses through these levels at fixed intervals of 100 steps, regardless of model performance. This represents traditional curriculum learning with static scheduling.

Adaptive Curriculum: Problem difficulty is adjusted dynamically based on observed model performance. Every 200 steps, we update our difficulty prediction model using recent performance data and select problems predicted to achieve approximately 50% success rate.

Sampling Adaptive Curriculum: Problem difficulty is estimated through brute-force solution analysis rather than performance feedback. We generate problems, solve them exhaustively, and use solution characteristics to predict difficulty via the same linear regression framework, then select problems targeting 50% predicted success rate.

4.3 Evaluation Protocol

We evaluate all models on the held-out evaluation set every 50 training steps to track learning progress. For each evaluation, we generate a single completion per problem and measure the percentage of problems solved correctly. This provides a consistent metric for comparing curriculum approaches throughout training.

The evaluation problems span the same difficulty range as the training problems but are completely separate to ensure we measure generalization rather than memorization. We also track additional metrics including mean reward, reward variance, and success rate distribution to understand training dynamics beyond final performance.

5 Results

Our experimental evaluation reveals important insights about curriculum learning in sparse reward environments, demonstrating both the potential benefits and practical considerations of different scheduling approaches. While adaptive curriculum methods show improved performance and training robustness, the overall gains are modest. The greatest practical benefit comes from the removal of the need to tune difficulty hyperparameters for the curriculum.

5.1 Quantitative Evaluation

Table 1 summarizes the final performance and training characteristics of each curriculum approach after 500 training steps. The random curriculum baseline achieves 59.0% success rate with stable training throughout. The fixed curriculum approach shows a complex pattern: strong initial performance reaching 52.9% success, followed by catastrophic degradation to 12.4% due to premature difficulty progression. Our adaptive curriculum achieves the highest final performance at 46.8% success while maintaining training stability.

Curriculum	Final Success	Peak Success	Stability
Random	59.0%	60.8%	High
Fixed	63.4%	64.1%	Low
Adaptive	64.5%	66.3%	High
Sampling	65.4%	65.4%	High

Table 1: Summary of curriculum learning results after 500 training steps. The adaptive approach provides the best combination of final performance and training stability, with approximately 10% faster convergence compared to random sampling.

Figure 1 illustrates the complete training dynamics for all three approaches. The curves reveal that while final performance differences are moderate, the learning trajectories can differ significantly. The adaptive and random curriculum demonstrate more efficient learning in the initial phase, with the fixed learning curriculum catching up around step 150. The adaptive sampling algorithm for curriculum generation shows great promise, but learns the slowest of all approaches. Further refinement of the sampling might show greater benefits, but an exploratory investigation of a single training run showed similar performance at 1000 steps and 500 steps, indicating that the model had converged.

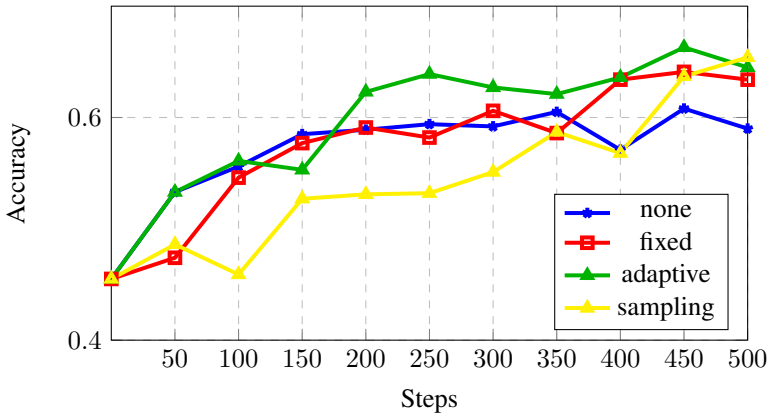

Figure 1: Training curves showing evaluation success rates over 500 training steps. The adaptive curriculum (green) achieves faster convergence than random sampling (blue) and $\sim 9\%$ better final performance. The fixed curriculum (red) converges more slowly than either fixed or adaptive curricula, partially because conservative difficulty hyperparameters were neccessary to avoid collapse. The sampling curriculum (yellow) shows promise, but current implementation selects too broad a band of difficulties due to implementation details.

5.2 Qualitative Analysis

The training curves reveal distinct characteristics for each curriculum approach that provide insights into the underlying learning dynamics. The random curriculum exhibits steady improvement until converging at a success rate of around 60%.

The fixed curriculum shows slow initial growth as the difficulty range had to be lowered substantially after initial hyperparameters exhibited reward starvation and model collapse. However, the model converges on a success rate substantially higher than random sampling. This highlights the challenge of manual hyperparameter tuning in curriculum learning systems.

Both adaptive approaches maintain steady improvement by automatically calibrating problem difficulty to model capabilities. The performance-based adaptive curriculum shows several phases of accelerated improvement where the difficulty prediction model successfully identifies optimal problem distributions. The sampling adaptive approach demonstrates similar stability while achieving slightly higher final performance, suggesting that solution-structure-based difficulty estimation may provide more consistent difficulty calibration.

The training efficiency and robustness benefits are more compelling than raw performance differences. The adaptive approaches eliminate the need for manual hyperparameter tuning to find optimal difficulty progression schedules, providing a more robust training framework that maintains performance across different configurations.

Perhaps most importantly, the results demonstrate that curriculum learning can significantly harm training when poorly implemented. The fixed curriculum's catastrophic failure serves as a cautionary tale about the brittleness of predetermined difficulty schedules in sparse reward environments. This finding has practical implications for curriculum learning deployment in problem domains with sparse rewards, suggesting that adaptive approaches may be necessary not just for performance optimization but for basic training stability.

6 Discussion

Our results demonstrate both the potential and limitations of adaptive curriculum learning for small language models in sparse reward environments. While the observed improvements are modest, the findings raise important questions about the fundamental mechanisms underlying curriculum learning and the practical requirements for implementing such systems in real-world scenarios.

6.1 The Difficulty Estimation Challenge

A critical assumption underlying our approach is the ability to meaningfully categorize and compare problems by difficulty. The success of our linear regression predictors suggests that countdown problems do exhibit systematic patterns in their difficulty structure. Problems requiring division operations, those with prime targets, and those necessitating deep operation trees consistently challenge the model more than problems solvable through simple addition or multiplication.

However, the modest performance gains raise questions about the completeness of our difficulty characterization. The most important aspects of problem difficulty may not be captured by our feature extraction approach, or difficulty may be more context-dependent than our framework assumes. Furthermore, the overhead of maintaining accurate difficulty estimates may offset the benefits of curriculum learning for many practical applications.

6.2 Mechanisms of Curriculum Learning Benefits

The modest but consistent improvements from adaptive curriculum scheduling suggest curriculum learning benefits arise primarily from gradient signal optimization rather than hierarchical skill building. Our adaptive approach succeeds by avoiding situations where all RLOO completions receive identical rewards, maintaining approximately 50% success rates to ensure consistent variance in advantage estimates.

This mechanism suggests curriculum learning benefits may be largely procedural rather than substantive. The improvements arise not from optimal skill sequencing but from avoiding pathological training dynamics that occur when problem difficulty is poorly calibrated to model capabilities. This

interpretation is supported by our biggest success being prevention of catastrophic collapse rather than achieving dramatic performance improvements.

Once we avoid uniform reward scenarios, further optimization of problem selection provides diminishing returns. The remaining performance differences between our adaptive approach and random sampling may simply reflect inevitable variance in problem difficulty within any reasonable problem distribution.

6.3 Implications for Curriculum Learning Research

Our findings suggest that curriculum learning benefits may be more limited than commonly assumed, particularly for tasks without clear hierarchical skill dependencies. The moderate improvements we observe, despite careful difficulty calibration and adaptive scheduling, indicate that curriculum learning may offer diminishing returns for certain problem domains.

The success of our adaptive approach in preventing training collapse suggests that curriculum learning may be most valuable as a safeguard against training instabilities rather than as a performance optimization technique. In sparse reward environments where poorly calibrated difficulty can cause catastrophic failure, adaptive curriculum scheduling provides insurance against training disasters even if it does not dramatically improve final performance.

7 Conclusion

This work investigates adaptive curriculum learning for small language models in sparse reward reinforcement learning environments, using the countdown numbers game as a testbed for mathematical reasoning tasks. Our primary contribution is demonstrating that adaptive curriculum scheduling provides modest but consistent performance improvements while eliminating the need for manual hyperparameter tuning of difficulty progression schedules. Comparing four curriculum strategies, we find that adaptive approaches achieve 64.5-65.4% final success compared to 59.0% for random sampling and 63.4% for fixed curriculum.

The most important finding is that adaptive curriculum scheduling provides training robustness without requiring domain expertise or extensive hyperparameter search. While fixed curriculum approaches can achieve strong performance when properly tuned, they require careful calibration and can suffer catastrophic collapse when difficulty progression outpaces model capabilities. Our analysis suggests that curriculum learning benefits arise primarily from maintaining optimal gradient signal variance rather than from hierarchical skill building, indicating that curriculum learning may be most valuable as a training stabilization technique rather than a pure performance optimizer. Future work should investigate whether curriculum learning benefits scale with model size or emerge more clearly in domains with stronger hierarchical structure, as understanding when curriculum learning provides meaningful benefits will be crucial for its effective application in practical machine learning systems.

8 Team Contributions

- **Nathaniel Voorhies** – Whole Project

Changes from Proposal The issue of reward starvation necessitated making an adaptive curriculum generator.

The issue of transfer learning from the generated problems to the real datasets was attempted to be solved with the sampling curriculum generator, but still needs some fine tuning.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning* (Montreal, Quebec, Canada) *(ICML '09)*. Association for Computing Machinery, New York, NY, USA, 41–48. https://doi.org/10.1145/1553374.1553380

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu,

Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948 [cs.CL] `https://arxiv.org/abs/2501.12948`

Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D. Goodman. 2025. Cognitive Behaviors that Enable Self-Improving Reasoners, or, Four Habits of Highly Effective STaRs. arXiv:2503.01307 [cs.CL] `https://arxiv.org/abs/2503.01307`

Marwa Naïr, Kamel Yamani, Lynda Said Lhadj, and Riyadh Baghdadi. 2024. Curriculum Learning for Small Code Language Models. arXiv:2407.10194 [cs.LG] `https://arxiv.org/abs/2407.10194`

Anikait Singh. 2025. Asap7772/cog$_b ehav_a ll_s trategies$.

Zhiheng Xi, Wenxiang Chen, Boyang Hong, Senjie Jin, Rui Zheng, Wei He, Yiwen Ding, Shichun Liu, Xin Guo, Junzhe Wang, Honglin Guo, Wei Shen, Xiaoran Fan, Yuhao Zhou, Shihan Dou, Xiao Wang, Xinbo Zhang, Peng Sun, Tao Gui, Qi Zhang, and Xuanjing Huang. 2024. Training Large Language Models for Reasoning through Reverse Curriculum Reinforcement Learning. [arxiv]2402.05808 [cs.AI] `https://arxiv.org/abs/2402.05808`

A Additional Experiments

Before the final hyperparameter values were finalized, preliminary runs with the fixed schedule exhibited dramatic model collapse, often apparently being pulled by the KL penalty into strange behavior where they would output nonsensical text. Once the difficulty was lowered, this rewards starvation no longer occurred, but there were substantial penalties in learning rate, which might be fixed by further tuning the minimum and maximum difficulty values.

Experiments with the sampling adaptive curriculum are hampered by the code's immaturity, especially as the filtering is too loose, allowing a wider range than optimal of problems to be accepted. We should follow up with a more compute intensive but more precise selection of problems determined to be optimal by the model.

Follow on work might want to explore the use of more powerful models for the difficulty estimation. There's a balance between need for data and difficulty model accuracy, and we are likely on the side of having too simple a model for the problem domain. Also, many of the features would benefit from

being turned into one-hot vector encodings, rather than a real value. This was avoided because of the fear of weakening the model with too many dimensions and not enough data.

B Implementation Details

Some implementation details were made in the interest of ease of implementation rather than elegance. For instance, the difficulty estimation makes extra calls to the model and solution evaluator, adding roughly 10% overhead to training. 3 It would have been more efficient (although the difficulty data would become slightly stale) to track all $RLOO_k$ attempts and the feature vectors of the related problems and use this data to build a dificulty estimate.2

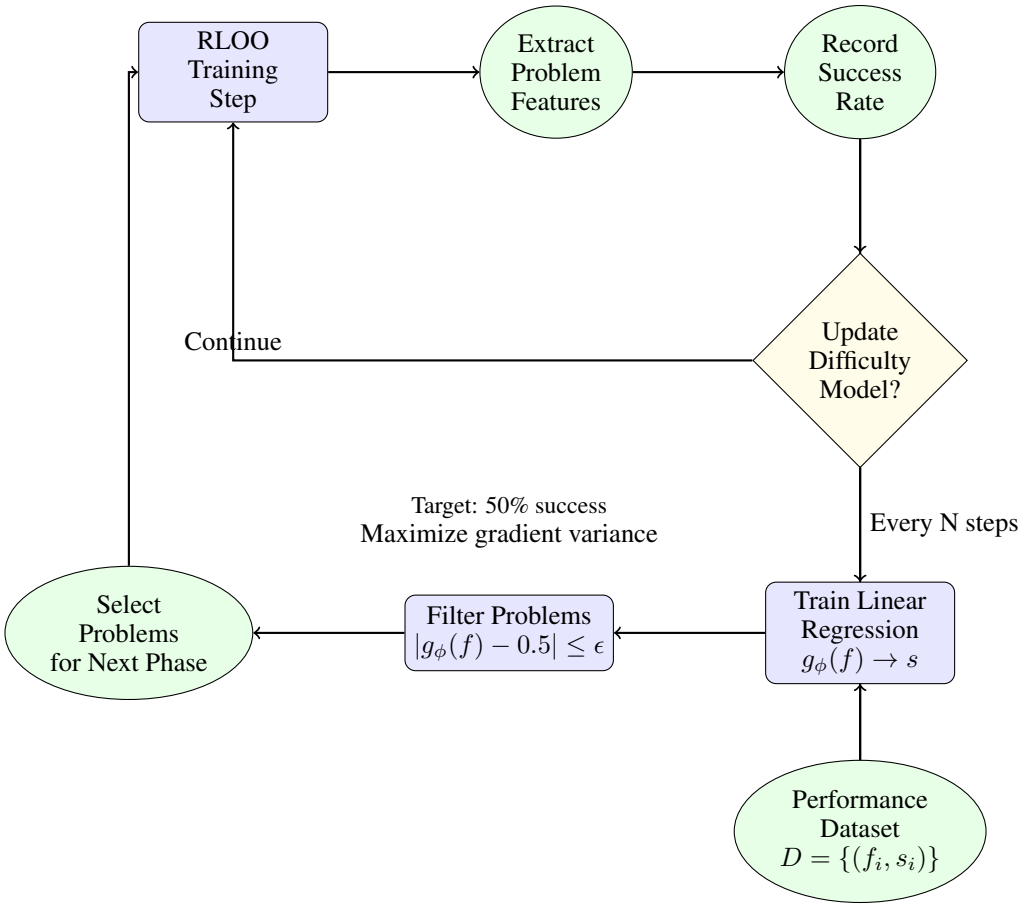

Figure 2: Adaptive curriculum training flow. The system continuously monitors model performance to update difficulty predictions and select problems targeting 50% success rates.

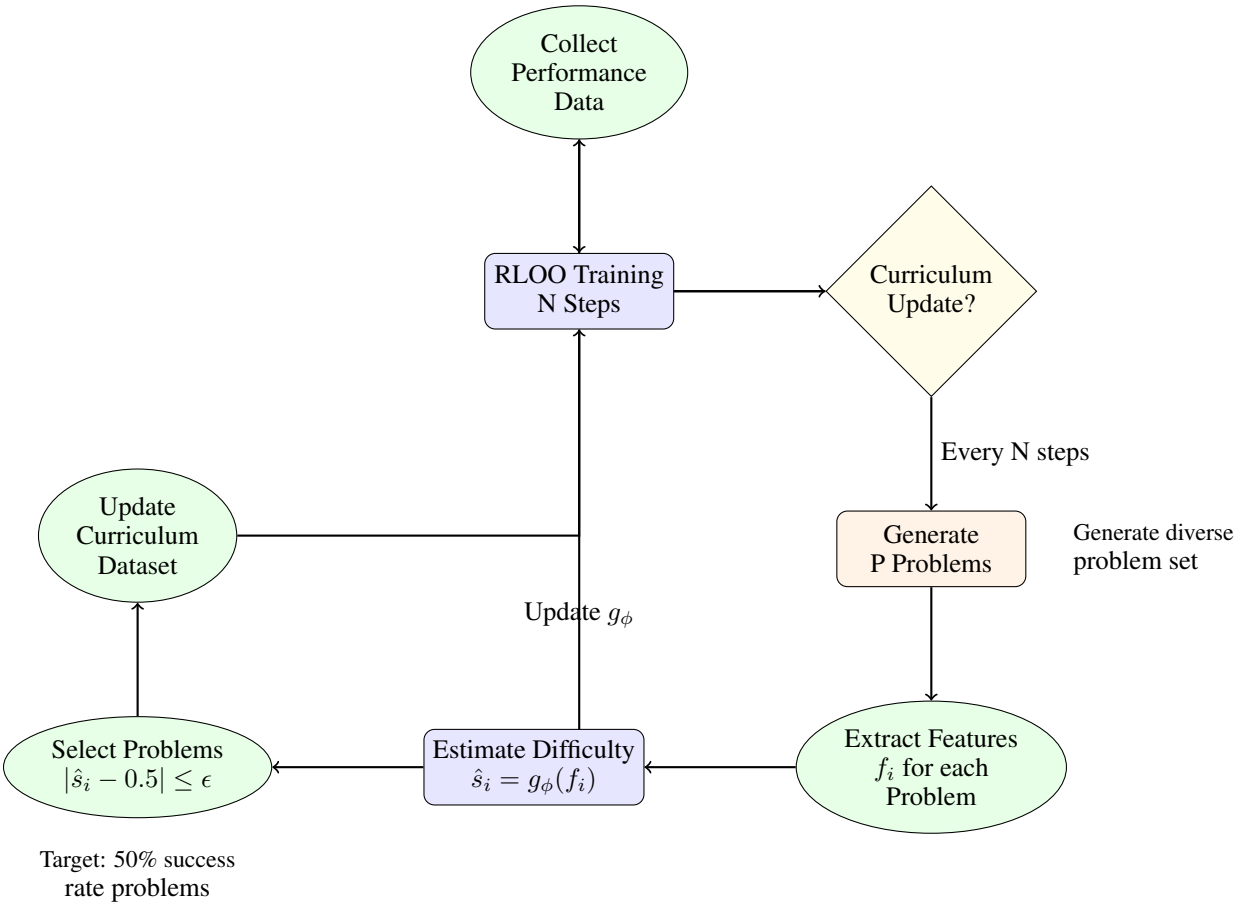

Figure 3: Batch curriculum update flow. Every N training steps, the system generates P new problems, estimates their difficulties, and selects those targeting 50% success rate for the next training phase.