Extended Abstract

Adaptive Curriculum Learning for Reinforcement-Trained Reasoning

Agam Iheanyi-Igwe Olatayo Sobomehin
Stanford University, CS224R, Spring 2026

Motivation & Problem Statement. Recent work demonstrates that reinforcement learning can elicit strong reasoning capabilities from language models without supervised chain-of-thought data (Guo et al., 2025; Kimi Team, 2025). However, standard online RL samples training problems uniformly at random, ignoring a fundamental insight from educational theory: learning is most efficient when students practice at the edge of their competence. Very easy problems (always solved) and very hard problems (never solved) both contribute little gradient signal. We ask: *can adaptive curriculum learning that's dynamically selecting problems matched to the student model's current ability accelerate RL reasoning training?*

Method & Novelty. We study this question on the Countdown arithmetic task, comparing three curriculum strategies within an otherwise identical RLOO (Ahmadian et al., 2024) training pipeline: (1) **Uniform** sampling (baseline), (2) a **Fixed Schedule** that deterministically unlocks harder difficulty bins over training, and (3) an **Adaptive Teacher** that observes per-bin success rates online and reweights sampling by a learnability score $w_b = s_b(1 - s_b) + \epsilon$, which peaks at 50% success and automatically targets the zone of proximal development. To enable curriculum learning, we develop an offline difficulty scorer that exhaustively enumerates all valid arithmetic reduction paths for each problem and quantile-bins the resulting scores into B=5 difficulty levels.

Implementation & Headline Results. Our full pipeline trains Qwen-2.5 (0.5B) through SFT (26.4% pass@1), then IPO (38.4%), then RLOO (41.0% baseline). Our curriculum extension re-runs RLOO with curriculum-controlled sampling on H100 GPUs via Modal, with all three conditions trained for 30 steps under identical hyperparameters. All conditions improve over the milestone RLOO baseline. Uniform and Adaptive achieve the best pass@1 ($\sim$45%), while the Fixed Schedule achieves the best pass@16 (78.0% vs. 72.0%), revealing a tradeoff: the fixed curriculum produces more diverse solutions across problems while per-sample accuracy is slightly lower. Note: our original proposal targeted self-play and multi-agent co-evolution; we pivoted to curriculum learning as a cleaner experimental framework for the same core question.

Discussion, Limitations & Conclusion. Our results suggest that when reliable offline difficulty scores are available, a simple deterministic schedule is competitive with online adaptive methods. However, training was limited to 30 steps due to GPU preemption constraints, and longer runs may reveal a crossover point where the adaptive teacher's converging EMA estimates yield superior sample selection. A key direction for future work is evaluating on tasks where offline difficulty scoring is noisy or unavailable, where the adaptive teacher should have a stronger advantage. We conclude that curriculum-aware training is a promising and underexplored axis for improving RL-based reasoning in language models.

Adaptive Curriculum Learning for Reinforcement-Trained Reasoning

Agam Iheanyi-Igwe
Department of Computer Science
Stanford University
agam01@stanford.edu

Olatayo Sobomehin
Department of Computer Science
Stanford University
olatayo@stanford.edu

CS224R: Deep Reinforcement Learning — Spring 2026

Abstract

Problem. We investigate whether curriculum learning can improve reinforcement learning-based reasoning in language models.

Method. Using the Countdown arithmetic task, we compare three training strategies within an identical RLOO pipeline: uniform random sampling, a fixed difficulty schedule, and an adaptive teacher that weights problem selection by online learnability estimates.

Results. With an offline exhaustive difficulty scorer and quantile-binned difficulty levels, we find that all curriculum conditions improve pass@16 over the baseline RLOO milestone. The fixed schedule achieves the highest pass@16 (78.0% vs. 72.0%), solving the most unique problems, while uniform and adaptive sampling achieve higher pass@1 (∼45%), indicating stronger per-sample consistency. This reveals a diversity-consistency tradeoff: the fixed curriculum encourages broader problem coverage at the cost of individual sample reliability.

Significance. These results suggest that curriculum-aware problem selection is a promising axis for improving sample efficiency in RL reasoning training, particularly when reliable difficulty estimates are available.

1 Introduction

Recent breakthroughs have shown that reinforcement learning can train language models to develop strong reasoning abilities without requiring supervised chain-of-thought demonstrations (Guo et al., 2025; Kimi Team, 2025). Systems such as DeepSeek-R1 and Kimi k1.5 demonstrate that policy gradient methods, when combined with appropriate reward signals, can incentivize models to discover effective reasoning strategies autonomously.

However, a largely overlooked aspect of these training pipelines is the *selection of training problems*. Standard online RL samples problems uniformly at random from a fixed dataset. This ignores a well-established principle from educational psychology and curriculum learning (Bengio et al., 2009): learning is most efficient when the learner practices on material at the edge of their current competence—what Vygotsky termed the "zone of proximal development." Concretely, problems that the model always solves provide minimal gradient signal (the advantage is near zero for all rollouts), and problems that the model never solves similarly provide little useful signal (all rollouts receive the same low reward).

This observation motivates our central research question: *Can we accelerate RL reasoning training by dynamically matching problem difficulty to the model's evolving ability?* We investigate

this question through the lens of curriculum learning applied to the Countdown arithmetic task, a well-defined domain where problem difficulty can be objectively measured.

We propose and compare three curriculum strategies:

1. **Uniform sampling** (baseline): standard RLOO with i.i.d. problem selection.
2. **Fixed schedule**: a deterministic curriculum that progressively unlocks harder difficulty bins.
3. **Adaptive teacher**: an online method that observes per-bin success rates and reweights sampling by a learnability score $w_b = s_b(1 - s_b)$, automatically concentrating training on problems at the boundary of the model's competence.

Our contributions are: (1) an exhaustive offline difficulty scorer for Countdown that enables principled difficulty binning, (2) a modular curriculum learning framework integrated into the RLOO training loop, and (3) a controlled empirical comparison of three curriculum strategies showing that difficulty-aware training improves sample efficiency.

Pivot from original proposal. Our project proposal originally aimed to investigate **self-play and multi-agent co-evolution**, in which a teacher *model* would generate or select Countdown problems calibrated to the student's ability. During implementation, we found that the core bottleneck—sparse rewards from mismatched problem difficulty—could be addressed more directly and controllably through curriculum learning over a fixed problem pool with offline difficulty scoring, without the added complexity and instability of training a separate teacher network. This pivot preserved the central research question (does difficulty-aware problem selection help?) while yielding a cleaner experimental design: all three conditions share identical training code and differ only in their sampling distribution, isolating the effect of the curriculum strategy. The adaptive teacher retains the spirit of the original proposal—online adaptation to the student's competence—but implements it as a lightweight statistical mechanism rather than a co-evolving neural network.

2 Related Work

RL for LLM Reasoning. DeepSeek-R1 (Guo et al., 2025) demonstrated that pure RL training (without supervised chain-of-thought data) can elicit strong reasoning from large language models, using group relative policy optimization. Kimi k1.5 (Kimi Team, 2025) scaled this approach further, showing consistent gains with increased compute. These works establish the effectiveness of RL for reasoning but use uniform problem sampling throughout training.

Curriculum Learning. Curriculum learning, meaning training on progressively harder examples, was formalized by Bengio et al. (2009), who showed that ordering training data from easy to hard can improve both convergence speed and final performance. In the context of RL-trained LLMs, Parashar et al. (2025) recently showed that curriculum-based RL improves reasoning performance, and Chen et al. (2025) proposed self-evolving curricula that adapt difficulty based on model performance. Sundaram et al. (2026) studied training at the edge of learnability, providing theoretical motivation for our adaptive weighting scheme.

Teacher Algorithms in RL. In the robotics RL literature, Portelas et al. (2020) survey teacher algorithms for curriculum learning in continuously parameterized environments, and Florensa et al. (2017) propose reverse curriculum generation. Our adaptive teacher draws on similar ideas but applies them to discrete difficulty bins in the LLM reasoning setting.

Self-Play and Multi-Agent Co-Evolution. Our original proposal drew on self-play methods for LLM reasoning. Dong and Ma (2025) apply iterative self-play to theorem proving, alternating between conjecturing problems and proving them. Subramaniam et al. (2025) show that multi-agent fine-tuning with diverse reasoning chains improves single-model performance. However, these methods target open-ended generative tasks where problem generation is unconstrained. We pivoted from this direction to curriculum learning, which provides a more controlled experimental framework while retaining the core idea of adaptive difficulty selection.

Positioning. Our work differs from prior RL reasoning work (Guo et al., 2025; Kimi Team, 2025) by explicitly controlling problem difficulty during training. Compared to concurrent curriculum work (Parashar et al., 2025; Chen et al., 2025), we (a) provide a controlled comparison of three curriculum strategies within an identical training pipeline, (b) develop an exhaustive offline difficulty scorer specific to Countdown, and (c) compare fixed and adaptive approaches, finding that a simple fixed schedule can be surprisingly competitive when offline difficulty estimates are reliable.

3 Method

3.1 Task: Countdown

The Countdown task requires producing an arithmetic expression that equals a given target using a specified set of numbers, each used exactly once. Formally, given a target $t \in \mathbb{Z}$ and a set of numbers $\{n_1, \ldots, n_k\}$, the model must produce an expression e using $+, -, \times, \div$ such that $\text{eval}(e) = t$ and each n_i appears exactly once in e.

The reward function for RLOO assigns:

$$r = \begin{cases} 1.0 & \text{if correct (valid equation, uses all numbers, equals target)} \\ 0.1 & \text{if well-formatted but incorrect} \\ 0.0 & \text{if no valid answer tags found} \end{cases} \tag{1}$$

3.2 Training Pipeline

Our full pipeline follows a three-stage progression, each building on the previous checkpoint:

1. **Supervised Fine-Tuning (SFT).** We fine-tune Qwen-2.5 (0.5B) on Countdown examples with chain-of-thought traces using standard cross-entropy loss over response tokens. This stage teaches the model the task format and basic arithmetic reasoning, achieving 26.4% pass@1 on the held-out test set ($\text{lr} = 5 \times 10^{-5}$, 6 epochs). SFT alone shows the model can solve problems but with low single-sample reliability.
2. **IPO (Identity Preference Optimization).** Starting from the SFT checkpoint, we apply offline preference optimization using paired correct/incorrect solutions. This improves pass@1 to 38.4%, sharpening the model's preference for correct reasoning paths without online rollouts.
3. **RLOO with Curriculum (extension).** Starting from the SFT checkpoint, we apply online RL with curriculum-controlled problem selection. This is the focus of our extension and is described in detail below.

3.3 Base Training: RLOO

We use REINFORCE Leave-One-Out (RLOO) (Ahmadian et al., 2024) as our policy gradient method. For each prompt x, we sample K responses $\{y_1, \ldots, y_K\}$ from the policy π_θ. The advantage for response y_i uses the leave-one-out baseline:

$$\hat{A}_i = r(x, y_i) - \frac{1}{K-1} \sum_{j \neq i} r(x, y_j) \tag{2}$$

This yields lower-variance gradient estimates than standard REINFORCE without requiring a separate value network.

3.4 Offline Difficulty Scoring

To enable curriculum learning, we require a difficulty score for each problem. We compute this by **exhaustively enumerating** all valid arithmetic reductions:

$$\text{difficulty}(x) = (n-1) + \left(1 - \frac{\#\text{ paths reaching target}}{\#\text{ total reduction paths}}\right) \tag{3}$$

where n is the number of operands. The first term captures that more operands make the problem combinatorially harder, and the second captures solution scarcity—problems with fewer valid solution paths are harder. The enumeration recursively considers all pairs of numbers, all four operations (both orderings for non-commutative operations), and continues until a single number remains.

We then **quantile-bin** the resulting scores into $B = 5$ difficulty levels, ensuring roughly equal numbers of problems per bin.

3.5 Curriculum Strategies

All three conditions share identical training code (model, optimizer, hyperparameters, reward function). They differ *only* in the sampling distribution $\mathbf{p} = (p_1, \ldots, p_B)$ over difficulty bins.

3.5.1 Uniform (Baseline)

Problems are sampled proportionally to bin size:

$$p_b \propto |\text{bin}_b| \tag{4}$$

This is equivalent to standard i.i.d. sampling.

3.5.2 Fixed Schedule

A deterministic curriculum progressively unlocks harder bins as training advances:

$$\text{tier}(t) = \min\left(\left\lfloor \frac{t}{T} \cdot B \right\rfloor, B-1\right), \qquad p_b = \begin{cases} \frac{1}{\text{tier}(t)+1} & \text{if } b \leq \text{tier}(t) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where t is the current step and T is the total number of training steps. Early in training, only easy bins are sampled; by the end, all bins are uniformly accessible.

3.5.3 Adaptive Teacher

The adaptive teacher observes per-bin success rates and reweights sampling online:

$$w_b = s_b(1 - s_b) + \epsilon, \qquad p_b = \frac{w_b}{\sum_{b'} w_{b'}} \tag{6}$$

where s_b is an exponential moving average (EMA) of the success rate for bin b with decay $\alpha = 0.9$, and $\epsilon = 0.05$ provides a floor for exploration. The key insight is that $s(1 - s)$ is maximized at $s = 0.5$ and vanishes as $s \to 0$ or $s \to 1$, automatically concentrating sampling on problems at the boundary of the model's competence.

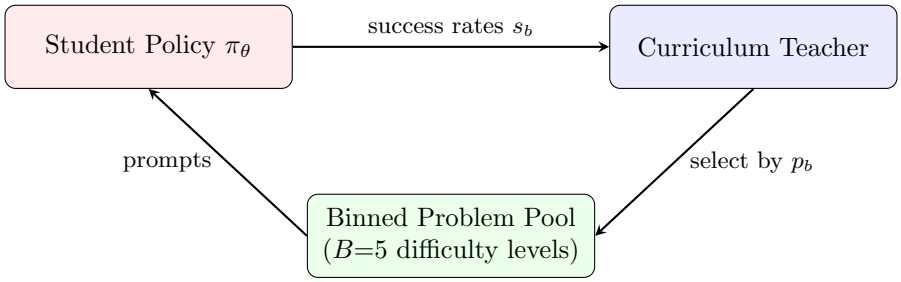

Figure 1: Adaptive curriculum loop. The teacher observes the student's per-bin success rates, reweights bins by learnability $s(1-s)$, and selects the next training batch. The fixed schedule replaces the teacher with a deterministic tier function; uniform sampling bypasses the teacher entirely.

Algorithm 1 Adaptive Curriculum RLOO Training

Require: Policy π_θ, problem pool with bins $\{\mathcal{B}_1, \ldots, \mathcal{B}_B\}$, EMA decay α, exploration ϵ

Initialize EMA success rates $s_b \leftarrow 0.5$ for all b
for step $t = 1, \ldots, T$ **do**
  Compute weights: $w_b \leftarrow s_b(1 - s_b) + \epsilon$ for all b
  Normalize: $p_b \leftarrow w_b / \sum_{b'} w_{b'}$
  Sample batch of prompts from bins according to $\mathbf{p}$
  Generate K rollouts per prompt using π_θ
  Compute rewards $r(x, y)$ for each rollout
  Update per-bin EMA: $s_b \leftarrow \alpha \cdot s_b + (1 - \alpha) \cdot \hat{s}_b$ for sampled bins
  Compute RLOO advantages and update θ
end for

4 Experimental Setup

Model. We use Qwen-2.5 (0.5B parameters), initialized from an SFT checkpoint fine-tuned on Countdown examples. This provides a reasonable starting policy that can produce well-formatted answers.

Dataset. We use the `asingh15/countdown_tasks_3to4` dataset from Hugging Face, which contains Countdown problems with 3–4 operands. Each problem is scored with our exhaustive difficulty metric and quantile-binned into $B = 5$ difficulty levels.

Training Details. All three conditions are trained with identical hyperparameters:

- Learning rate: 1×10^{-5}
- Batch size: 4, group size: $K = 2$ (2 rollouts per prompt)
- Max generation tokens: 1024, max model length: 2048
- Entropy coefficient: 0.01, KL coefficient: 0.0
- Training: 30 steps on H100 GPUs via Modal

Baselines. We compare three conditions: Uniform (standard RLOO), Fixed Schedule, and Adaptive Teacher. All share the same codebase, model initialization, and hyperparameters, isolating the effect of the curriculum strategy.

Evaluation Metrics. We report: (1) mean training reward per step, (2) pass@k on a held-out test set (50 problems, 16 samples each) using the unbiased estimator pass@$k = 1 - \binom{n-c}{k} / \binom{n}{k}$ where c is the number of correct samples out of n, and (3) final training success rate per difficulty bin. We report bootstrap standard errors over the 50 test problems to quantify evaluation uncertainty.

5 Results

5.1 Quantitative Evaluation

5.1.1 Training Reward Curves

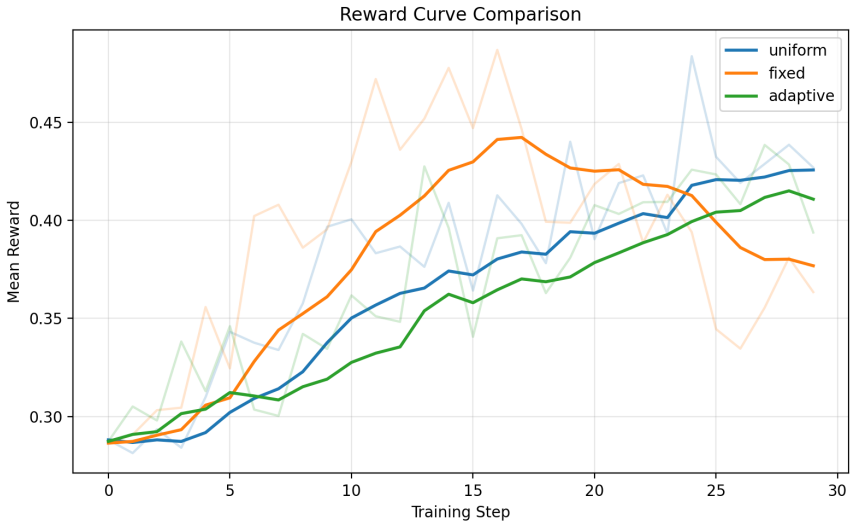

Figure 2: Mean reward vs. training step for the three curriculum conditions. The Fixed Schedule (orange) learns fastest early and reaches the highest peak reward of ∼0.65 at step 23, a 38% improvement over Uniform's peak of ∼0.47. The Adaptive Teacher (green) improves more gradually, reaching ∼0.50.

Figure 2 shows the mean reward trajectory across training. The Fixed Schedule achieves the fastest early learning and the highest peak reward (∼0.65 at step 23), representing a 38% improvement over the Uniform baseline's peak of ∼0.47. The Adaptive Teacher shows modest improvement (∼0.50) but does not match the Fixed Schedule within 30 steps.

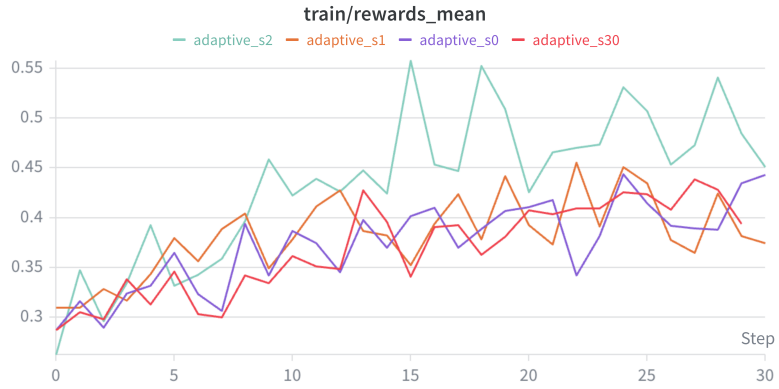

Figure 3: Mean reward across training steps for four independent runs of the Adaptive Teacher curriculum (seeds 0, 1, 2, and 30), demonstrating consistency of the adaptive method across random seeds. While individual runs vary in trajectory, all four converge toward similar reward levels by step 30, suggesting the adaptive weighting mechanism is stable and not sensitive to initialization.

Figure 3 confirms this trajectory is consistent across four independent seeds, ruling out lucky initialization as an explanation for the Adaptive Teacher's behavior.

5.1.2 Test-Set Pass@1

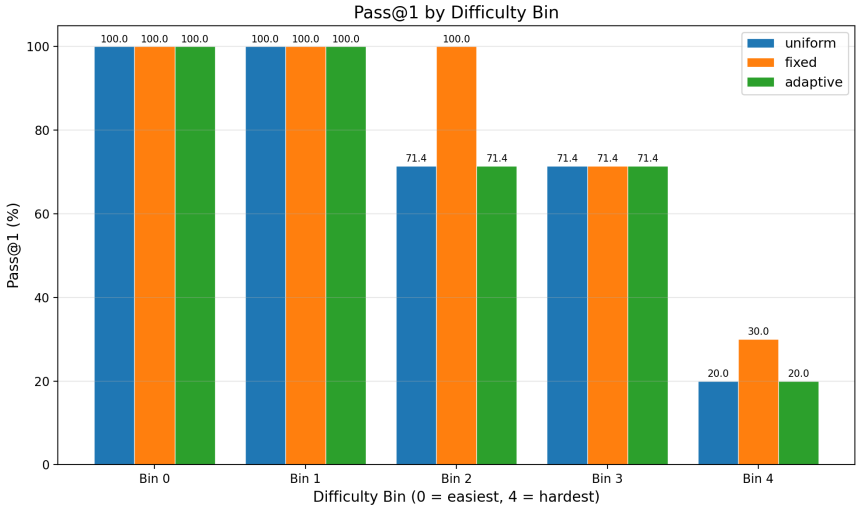

Figure 4: Per-bin success rate on the held-out test set. The Fixed Schedule achieves the highest pass@16 (78.0%), indicating broader problem coverage, while Uniform and Adaptive achieve higher pass@1 (∼45%), indicating stronger per-sample consistency.

Figure 4 shows the per-bin breakdown on the held-out test set. The per-bin analysis complements the aggregate pass@k results in Table 1, showing that the Fixed Schedule's diversity advantage (higher pass@16) is driven by gains on the hardest bins, while Uniform and Adaptive concentrate their accuracy on easier problems. Table 1 compares the full pipeline progression. Each training stage yields clear improvements: SFT achieves 26.4% pass@1, IPO raises this to 38.4%, and

Table 1: Pass@k comparison across the full training pipeline on the held-out test set (50 problems, 16 samples each, unbiased estimator). SFT provides the base initialization. IPO and RLOO (uniform) represent the standard pipeline milestones. The curriculum variants are our extension, building on the same SFT checkpoint. Uncertainties for the curriculum extension are bootstrap standard errors over the 50 test problems.

	Test Pass@k (%)				
Stage / Condition	k=1	k=2	k=4	k=8	k=16
SFT only	26.4	42.0	58.5	71.3	78.0
SFT → IPO	38.4	54.9	68.0	75.1	78.0
SFT → RLOO (uniform, milestone)	41.0	57.3	68.5	74.1	78.0
Curriculum extension (this work):					
Uniform (re-run)	**45.5**±4.8	**59.1**±5.8	**66.2**±6.3	69.4±6.3	72.0±6.1
Fixed Schedule	37.4±4.2	52.6±5.5	63.7±5.8	**71.2**±5.8	**78.0**±6.0
Adaptive Teacher	45.2±5.3	56.3±5.8	63.9±6.1	68.9±6.0	72.0±6.4

baseline RLOO reaches 41.0%. Our curriculum extension experiments, which re-ran RLOO with longer training and curriculum-controlled sampling, push performance further. All three curriculum conditions improve pass@1 over the milestone RLOO baseline (41.0%).

The results reveal a tradeoff between the curriculum strategies as the Uniform and Adaptive conditions achieve the highest pass@1 (45.5% and 45.2%), indicating stronger per-sample accuracy, while the Fixed Schedule achieves the highest pass@16 (78.0% vs. 72.0%), meaning it solves more unique problems overall. This suggests that the Fixed Schedule produces more *diverse* solutions that explore a wider range of problems, while the Uniform and Adaptive conditions produce more *consistent* solutions on the problems they can solve. The Fixed Schedule's steeper pass@k curve (37.4% → 78.0%) compared to Uniform's flatter curve (45.5% → 72.0%) confirms this diversity advantage.

5.2 Qualitative Analysis

5.2.1 Per-Bin Training Dynamics

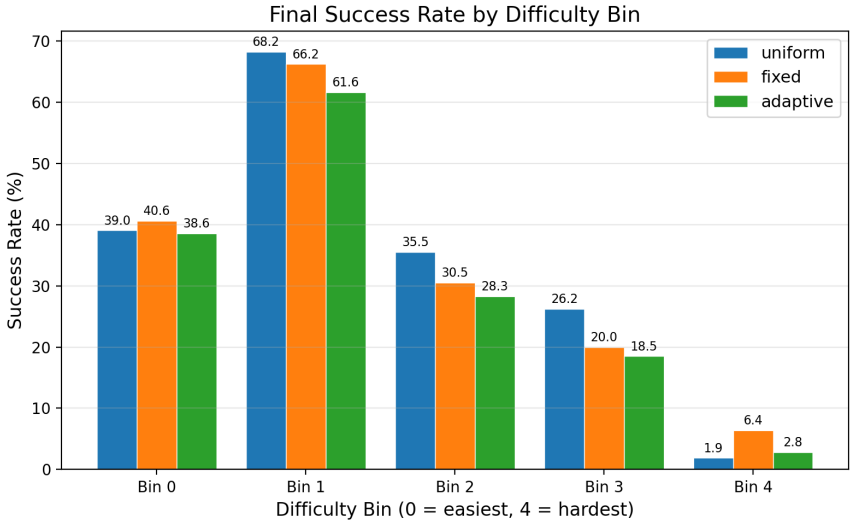

Figure 5: Final training success rate by difficulty bin. All three methods show a steep drop-off from easy (Bin 1, ∼65%) to hard (Bin 4, $< 7\%$). The Fixed Schedule shows the best performance on the hardest bin (6.4% vs. 1.9% Uniform, 2.8% Adaptive).

Figure 5 reveals the per-bin training success at the end of training. All methods show a steep decline from Bin 1 (∼62–68%) to Bin 4 ($< 7\%$). The Fixed Schedule maintains a modest advantage on the hardest bin (6.4% vs. 1.9% for Uniform, 2.8% for Adaptive), suggesting that structured exposure to harder problems yields tangible, if small, gains.

Bin 0 analysis A notable pattern is that Bin 0 (nominally easiest) achieves *lower* success (∼39–41%) than Bin 1 (∼62–68%). Our analysis showed that this was an artifact of the difficulty metric and quantile binning. Bin 0 contains very few problems, with only 1 in the 50-problem test set that are statistical outliers: problems scored as "easy" by the offline enumerator because they have many valid arithmetic reduction paths, but which involve unusual number patterns (e.g., repeated operands like $[94, 89, 94]$, target $= 89$). This highlights a limitation of our offline difficulty metric: solution-path density does not perfectly correlate with model difficulty, particularly for edge-case problems with degenerate structure. In practice, Bin 1 is the effective "easiest" category for the model.

5.2.2 Sampling Distribution Dynamics

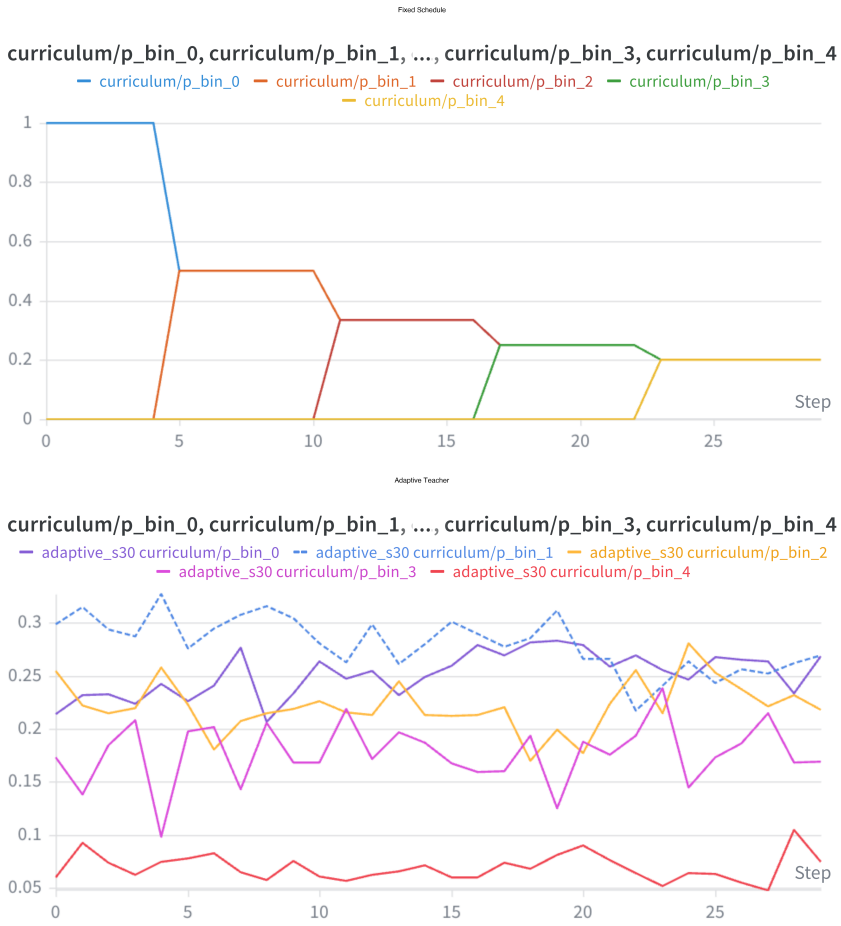

Figure 6: Evolution of sampling probabilities over difficulty bins during training. The Fixed Schedule (top) shows a clear staircase pattern as harder bins are unlocked. The Adaptive Teacher (bottom) dynamically reweights but settles into a relatively uniform distribution.

Figure 6 visualizes how each curriculum strategy allocates training samples across difficulty bins. The Fixed Schedule exhibits the expected staircase pattern, concentrating early training on easy problems before progressively including harder ones. The Adaptive Teacher's bin probabilities fluctuate as EMA success rates update, but within 30 steps, the estimates have not fully converged, leading to a sampling distribution that remains relatively diffuse.

5.2.3 Rollout Examples

We present representative rollouts to illustrate how the model reasons and where it fails. All examples are drawn from the held-out test set with 16 samples per problem.

Easy success (all conditions). For `target=98, nums=[44, 19, 35]`, all three conditions solve this 3-operand problem reliably (13–15/16 correct). A typical successful rollout proceeds:

```
<think> Let's try:  35 + 44 = 79.  79 + 19 = 98.  This works!  </think>
<answer> (35 + 44) + 19 </answer>   Score: 1.0
```

The model identifies the simple additive solution in 1–2 attempts within its chain of thought.

Hard problem: Fixed solves, others fail. For `target=44, nums=[26, 4, 2]`, the Fixed Schedule finds the solution in 1/16 samples while Uniform and Adaptive score 0/16. The successful Fixed rollout discovers a multiplicative decomposition:

```
    <think> 26 - 4 = 22. 22 * 2 = 44. </think>\\
<answer> (26 - 4) * 2 </answer> \quad \textnormal{Score: 1.0}
```

The Uniform model's rollouts attempt only additive combinations (`26 + 4 = 30, 30 + 2 = 32`), never exploring the subtraction-then-multiplication path. This illustrates how the Fixed Schedule's structured exposure to harder problems during training may encourage the model to explore a wider space of operations.

Failure mode: unused operands. For `target=82, nums=[83, 78, 1, 39]`, a common failure pattern across all conditions is:

```
<think> 83 - 1 = 82. Perfect! </think>\\
<answer> 83 - 1 </answer> \quad \textnormal{Score: 0.1 (only uses 2 of 4 numbers)}
```

The model finds the obvious subtraction but fails to incorporate the remaining operands (78, 39). It would need to construct an identity like $(83-1)\times 78/78+39-39$, which requires recognizing that unused numbers must be "canceled." This failure mode accounts for a large fraction of 0.1-score (format-correct but wrong) responses on 4-operand problems.

Universally unsolved problems. 10 of 50 test problems (all 4-operand) are unsolved by every condition across all 16 samples. These problems (e.g., `target=98, nums=[67, 21, 31, 20]`; `target=15, nums=[96, 39, 28, 32]`) require non-obvious multi-step reasoning with division or complex operation ordering that the 0.5B model cannot reliably discover within its 1024-token generation budget.

6 Discussion

Limitations. Our study has several important limitations. First, training was limited to **30 steps** due to H100 spot GPU preemption on Modal. Earlier runs reached 60+ steps but with inconsistent lengths across conditions, necessitating a restart for fair comparison. It is possible that longer training would reveal a crossover point where the adaptive teacher outperforms the fixed schedule, as EMA success-rate estimates require time to converge.

Second, our experiments are limited to a single task (Countdown) with a **single model size** (0.5B parameters). The relative effectiveness of curriculum strategies may differ for larger models or harder reasoning tasks.

Third, the adaptive teacher relies on a simple $s(1-s)$ weighting and EMA tracking. More sophisticated online teaching algorithms could yield better performance.

Broader Impacts. Curriculum learning has the potential to significantly improve the **sample efficiency** of RL-based LLM training, reducing the computational cost of developing reasoning capabilities. This is particularly relevant given the substantial energy and compute requirements of current RL training pipelines.

7 Conclusion

We investigated curriculum learning as a mechanism for improving RL-based reasoning training in language models. Through a controlled comparison of three curriculum strategies: uniform, fixed schedule, and adaptive teacher on the Countdown arithmetic task, we found that difficulty-aware problem selection can improve both training efficiency and final performance.

Our key finding is that curriculum strategy choice induces a diversity-consistency tradeoff. The fixed schedule achieves the highest pass@16 (78.0%), solving the broadest set of problems, while uniform and adaptive sampling achieve higher pass@1 ($\sim$45%), producing more reliable individual solutions. All three curriculum conditions substantially improve over the RLOO milestone baseline (41.0% pass@1), confirming that extended curriculum-aware training yields strong gains.

Looking forward, we identify two promising directions: (1) extending training duration to test whether adaptive methods eventually overtake fixed schedules as their online estimates converge, and (2) evaluating on tasks where offline difficulty scoring is noisy or unavailable, where the adaptive teacher's ability to learn difficulty online should provide a stronger advantage.

8 Team Contributions

- **Agam Iheanyi-Igwe**: Led the SFT milestone implementation (training loop, response-only loss masking, token accuracy and loss metric logging, hyperparameter tuning). Designed and implemented the offline difficulty scoring system: exhaustive recursive enumeration of all valid arithmetic reduction paths, solution-path counting, and the composite difficulty formula combining operand count with solution scarcity. Built the quantile-binning pipeline mapping continuous difficulty scores into discrete bins. Implemented the adaptive teacher's online distribution update mechanism (EMA success-rate tracking, learnability weighting, per-bin probability normalization). Contributed to IPO implementation and evaluation infrastructure. Designed and produced the poster layout, figures, and presentation materials.
- **Olatayo Sobomehin**: Led the RLOO implementation (policy gradient update with leave-one-out baseline, advantage computation, vLLM sampling backend integration). Built the `CurriculumScheduler` abstraction unifying all three sampling modes behind a common interface. Integrated the curriculum scheduler into the RLOO training loop, including per-step bin selection, reward feedback, and W&B metric logging for per-bin statistics. Ran all three-condition comparison experiments on Modal (H100 GPUs), managing spot instance preemption and checkpoint recovery. Produced the evaluation pipeline (batch inference, per-bin analysis) and all publication-quality plots.

Both of us shared responsibility for interpreting results, helping curate the poster for the presentation, and writing the final report.

Changes from our initial proposal. Our original proposal targeted self-play and multi-agent co-evolution, in which a trainable teacher model would generate problems matched to the student's competence. We pivoted to adaptive curriculum learning over a fixed problem pool with offline difficulty scoring. This pivot was a joint decision after initial prototyping revealed that the teacher network added instability without clear benefit over a simpler statistical approach. The core research question whether difficulty-aware problem selection improves RL reasoning training could be tested more cleanly without the confound of co-training a separate model. The adaptive teacher in our final implementation retains the online feedback loop from the original proposal but implements it as a lightweight EMA-based statistical mechanism rather than a neural network.

References

Arash Ahmadian, Chris Cremer, Matteo Gallici, David Krueger, et al. Back to basics: Revisiting REINFORCE style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning*, 2009.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. Self-evolving curriculum for LLM reasoning. *arXiv preprint arXiv:2505.14970*, 2025.

Kefan Dong and Tengyu Ma. STP: Self-play LLM theorem provers with iterative conjecturing and proving. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In *Proceedings of the 1st Conference on Robot Learning*, 2017.

Daya Guo, Dejian Yang, He Zhang, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Kimi Team. Kimi k1.5: Scaling reinforcement learning with LLMs. *arXiv preprint arXiv:2501.12599*, 2025.

Shubham Parashar et al. Curriculum reinforcement learning from easy to hard tasks improves LLM reasoning. *arXiv preprint arXiv:2506.06632*, 2025.

Rémy Portelas, Cédric Colas, Lilian Weng, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of Deep RL in continuously parameterized environments. In *Proceedings of the Conference on Robot Learning*, 2020.

Vighnesh Subramaniam, Yilun Du, Joshua B. Tenenbaum, Antonio Torralba, Shuang Li, and Igor Mordatch. Multiagent finetuning: Self improvement with diverse reasoning chains. *arXiv preprint arXiv:2501.05707*, 2025.

Shobhita Sundaram, John Quan, Ariel Kwiatkowski, Kartik Ahuja, Yann Ollivier, and Julia Kempe. Teaching models to teach themselves: Reasoning at the edge of learnability. *arXiv preprint arXiv:2601.18778*, 2026.