Forget Less, Solve More: Sequential Fine-Tuning with Adapter Shrinking for Math Word Problems

Gauri Toshniwal [1] **S R Balasundaram** [1]

Abstract

This work investigates a human-inspired sequential fine-tuning (SeqFT) method to improve the performance of resource-constrained large language models (LLMs) on math word problems. Instead of training on the entire dataset simultaneously, models are exposed to progressively harder tasks level by level, while earlier data is periodically reintroduced to mitigate catastrophic forgetting. In addition, a strategy called Progressive LoRA Rank Shrinking (PLRS) is proposed, which progressively reduces the LoRA rank at each stage to prevent the overwriting of parameters learned in earlier levels. Evaluations on the MATH dataset demonstrate that this approach consistently outperforms both parameter-efficient fine-tuning and naive multi-level training, yielding up to a **2%-7%** improvement in exact match accuracy. The study presents the effect of (1) repeated data exposure, (2) difficulty-based task ordering via SeqFT, and (3) PLRS. An analysis of problem-solving trajectories further reveals that PLRS facilitates retention of earlier skills in a multi-stage setup. These findings suggest that, beyond conventional data augmentation, carefully designed training schedules can significantly enhance math problem-solving capabilities in LLMs.

1. Introduction

Recent LLMs excel at a range of tasks, from text generation to code completion. Yet mathematical problem solving remains challenging, especially when smaller or resource-constrained models are used. Typical approaches for improving math performance center on **data augmentation** e.g., generating synthetic examples (Lu et al., 2024; Li et al., 2024) or applying specialized **tool integrations** to offload symbolic manipulation (Das et al., 2024; Gou et al., 2024).

Problem statement : How can we systematically train a resource constrained LLM to handle multi-level math tasks without relying on large-scale data augmentation? We turn to **SeqFT**, reminiscent of how humans are taught math: starting with basic tasks and progressively adding advanced ones. The challenge, however, is *catastrophic forgetting*: a model that masters Level 1 tasks might forget them after exposure to Level 2 or 3. To counteract this, we re-expose earlier data i.e., presenting Level 1 tasks together with Level 2 tasks, and so on. Yet, we observe that simply replaying old tasks can *harm* performance if not paired with a suitable adaptation strategy.

This work also leverages findings that **LoRA adapters** can incrementally update a model with less forgetting (Biderman et al., 2024). We push this further by shrinking LoRA rank at each stage, hypothesizing that smaller-rank adapters at higher difficulty levels help the model preserve earlier skills while still adding specialized parameters for advanced tasks.

Our main contributions:

- This work introduces **SeqFT + PLRS**, a three-component recipe: staged curriculum, selective replay, and rank-shrinking adapters.
- The method demonstrates **positive backward transfer**: the best variant lifts Level-1 accuracy from 42 % to **53 %** *after* learning harder material, rather than merely preserving it.
- The approach delivers **2%–7%** absolute Exact match (EM) gains on four open-source backbones (0.5B – 3B) *without* increasing base-model parameters.
- A fine-grained *error-trajectory* analysis is provided, including (lost / recovered / newly-solved) buckets showing how replay and PLRS interact to reduce forgetting while unlocking new successes.

Extensive ablations isolate the effect of (*a*) replay, and (*b*) progressive rank shrinking, revealing that only their *synergy* yields the final 20.32 % EM over direct fine-tuning.

[1]National Institute of Technology, Tiruchirappalli, India. Correspondence to: Gauri Toshniwal <gaurigst1970@gmail.com>.

The Second AI for MATH Workshop at the 42nd International Conference on Machine Learning, Vancouver, Canada. Copyright 2025 by the author(s).

2. Related Work

Research on enhancing large language models for robust mathematical reasoning spans multiple directions, from data augmentation and tool integration to curriculum-style continual learning and novel evaluation benchmarks. In this section, we outline key strategies explored in recent work:

Data Augmentation and Supervised Fine-Tuning. A prevalent strategy is to gather or synthesize more math-specific training data. Foundational datasets like MATH (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021) have established strong baselines, but many follow-up works focus on expanding these corpora to fill reasoning gaps. Methods such as question back-translation (Lu et al., 2024), query augmentation (Li et al., 2024), or multi-source instruction tuning (Yue et al., 2023) often boost final accuracy by supplying more diverse examples. (Tang et al., 2024) explore concept graph extraction and topic generation, (You et al., 2024) emphasize reorganization and rephrasing of existing items.

Instruction Tuning and Chain-of-Thought Prompts Instruction tuning aligns language models with more helpful objectives by structuring tasks as explicit instructions (Longpre et al., 2023; Chung et al., 2024; Wang et al., 2022). In practice, many math-focused methods incorporate *chain-of-thought* prompts (Wei et al., 2022) by requesting models to show their intermediate reasoning steps. In our work, we follow this style of prompt design: at training time, we instruct the model to ***solve the following problem and show all intermediate steps.*** (See Appendix Figures 3 and 4 for the full prompt templates.) Moreover, having the final solution enclosed in box simplifies exact-match evaluation, as we can reliably extract the model's final numeric or symbolic answer for scoring.

Tool Integration and Verification Mechanisms. Another branch of work moves beyond pure language modeling by connecting LLMs with external solvers or verification modules. Techniques introduced by (Das et al., 2024) and (Gou et al., 2024) integrate symbolic calculators or structured reasoning frameworks, enabling models to offload complex manipulations to specialized systems. This can yield higher accuracy and more interpretable solution steps. While these approaches leverage additional tooling, our study remains focused on rearranging existing data within a purely language-based pipeline.

3. Background

3.1. The MATH Dataset

All experiments are based on the MATH dataset (Hendrycks et al., 2021), which categorizes problems into difficulty Levels 1- 5, refer Table - 1 for data distribution. The dataset has encoded a problem's difficulty level from '1' to '5,' following Art of Problem Solving. A subject's easiest problems for humans are assigned a difficulty level of '1,' and a subject's hardest problems are assigned a difficulty level of '5.' Our Sequential approach exploits this labeling by training from lowest to highest level, step by step. The dataset's strong difficulty gradient makes it an ideal testbed for analyzing structured fine-tuning schedules.

Table 1. Train and Test Data Distribution Across Levels

LEVEL	TRAIN COUNT	TEST COUNT
LEVEL 1	564	437
LEVEL 2	1348	894
LEVEL 3	1592	1131
LEVEL 4	1690	1214
LEVEL 5	2304	1324
TOTAL	**7498**	**5000**

3.2. Challenges and Techniques in SeqFT

Catastrophic forgetting: In a multi-level setting (where an LLM is fine-tuned multiple times sequentially), focusing on advanced tasks can overshadow the simpler fundamentals from previous levels. This phenomenon, known as catastrophic forgetting, has been extensively studied in the continual learning literature. Instruction tuning effectively optimizes LLMs for downstream tasks, but in evolving real-world scenarios, continual adaptation is necessary and without proper mitigation, this leads to forgetting. Prior work has shown that sequentially learning multiple skills in LLMs leads to degradation of earlier abilities (Dong et al., 2024).

Considering the computational burden of large-scale fine-tuning, replay-based continual learning strategies are among the most effective and widely adopted solutions (Wang et al., 2024). Recent works like SSR (Self-Synthesized Rehearsal) (Huang et al., 2024) further validate that LLMs suffer from forgetting.

In our case, the model is not switching to a completely new task but it is solving the same task (math word problems) with increasing difficulty. This progressive setup creates a unique challenge: the model may forget earlier, simpler skills as it focuses on mastering more complex reasoning. As we empirically demonstrate in Table 4, exact-match accuracy on Level 1 problems drops significantly after training on higher levels, clearly highlighting this forgetting effect even within a single domain.

LoRA and Adapter Shrinking: LoRA (Hu et al., 2021) addresses the efficiency of fine-tuning by updating only a small subset of parameters through low-rank adapters inserted into attention and feed-forward modules. While

LoRA can reduce catastrophic forgetting by avoiding full-parameter overwrite, training across multiple rounds using a static LoRA rank still risks overshadowing earlier learned representations.

We propose Progressive LoRA Rank Shrinking (PLRS): reduce the LoRA rank at each subsequent stage, allowing the model to ”lock in” earlier knowledge in high-rank adapters while assigning smaller, focused capacity to later stages. This mitigates interference, preserves earlier skills, and ensures better retention throughout the sequential training process.

4. Sequential Fine Tuning Method

Algorithm 1 Sequential Fine-Tuning Algorithm

```
Require: Pretrained model $M$, difficulty levels $L_1 < L_2 < \cdots < L_n$, dataset $D$ sorted by difficulty, LoRA configurations $\{R_1, R_2, \ldots, R_n\}$
Initialize $M$ with LoRA($R_1$)
for $i = 1$ to $n$ do
    $D_i \leftarrow \{x \in D \mid \text{difficulty}(x) \leq L_i\}$
    if $i > 1$ then
        $D_{replay} \leftarrow$ sample($D_{1 \ldots i-1}$)
        $D_i \leftarrow D_i \cup D_{replay}$
        Update $M$ with LoRA($R_i$)
    end if
    Fine-tune $M$ on $D_i$ for $E$ epochs using chosen hyper-parameters
end for
return Trained model $M$
```

Difficulty Sequencing. Segment tasks by their difficulty level, from Level 1 (easiest) through Level 5 (hardest). At stage i, the model trains on $\leq$Level i. This ensures a human-like progression: basic then more advanced questions.

Data Re-exposure (Replay). To avoid forgetting fundamentals, we keep earlier-level data in the training set each time we move to a higher level. Formally, at stage i, the model sees

$$D_i = \{x \in D \mid \text{difficulty}(x) \leq i\}. \quad (1)$$

To manage replay volume, a fraction ρ of earlier tasks can optionally be sampled during replay. However, naive full replay can occasionally degrade final performance if the model over-fits on easy tasks.

PLRS We start large at $r_1 = 256$, for example, to allow broad capacity for initial learning. At each subsequent stage, we shrink it (e.g., $256 \rightarrow 128 \rightarrow 64 \rightarrow 32$). Old adapters remain loaded, while newly introduced adapters for advanced tasks have smaller rank, preventing them from overwriting everything. In Figure 1 we show a conceptual diagram.

Taken together, these three components form the basis of our final approach Algorithm 1.

5. Experiments

5.1. Setup and Hyper-parameters

Dataset. The official **MATH** dataset split is used, consisting of 7.5k training problems annotated with difficulty levels (1–5) and step-by-step solutions, and a 5k-item test set. Difficulty levels follow Art of Problem Solving (AoPS) conventions, with *AMC 8* questions typically at level 1 and *AIME* questions at level 5. Each solution includes a boxed final answer, allowing for *exact-match* (**EM**) evaluation after standardizing whitespace and minor formatting.

Training recipe. All runs employ 4-bit NF4 quantisation, LoRA injection, the prompt template in Figure 4, and deterministic seeds. We fine-tune for five epochs per stage; more passes hurt validation EM, so we keep early-stopping (patience=2). Batch size is 4; other hyper-parameters appear in Appendix Tables 6 and 7. Checkpoints and logs are pushed to the Hugging Face Hub for full reproducibility.

Compute Cost Considerations. Although each stage in our sequential pipeline is configured for 5 epochs, early stopping is applied, and it is observed that training often converges much sooner, the full PLRS pipeline requires approximately 9000 training steps, compared to 4000 steps for the direct baseline. Notably, naively extending the baseline to 9000 steps leads to overfitting (lower validation EM), highlighting that our gains stem from structured training and rank allocation not just longer compute. All experiments were conducted using two NVIDIA Tesla T4 GPUs (15 GB each)

Models. We ablate on **LLaMA 3.2 1B** and replicate the best variant on **LLaMA 3.2 3B**, **Qwen2.5-Math 1.5B**, and **Qwen 2 0.5B** to test generality across scale and architecture.

LoRA schedules. Single-pass baselines always use rank=32. Our PLRS curriculum starts at $r = 256$ and shrinks $256 \rightarrow 128 \rightarrow 64 \rightarrow 32 \rightarrow 32$.

Capacity control. PLRS retains all adapter slices, so its total trainable parameters exceed a plain rank-32 run. To isolate *capacity scheduling* from mere size, we train a **rank-256 direct baseline**. Table 2 shows this heavy baseline *under-performs* rank-32 SFT, confirming that our gains stem from staged allocation plus replay, not just more parameters.

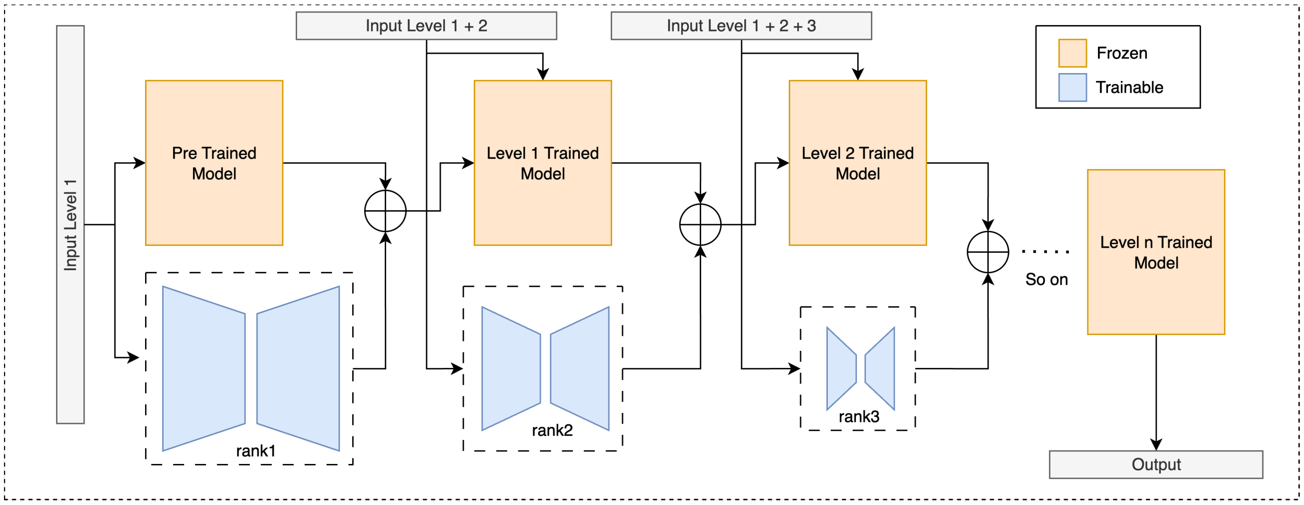

Figure 1. An illustration of PLRS
We begin with a large LoRA rank at Level 1, then load that adapter and add a smaller-rank adapter for Level 2, re-exposing old data and so on.

Table 2. Parameter-matched single-pass baselines

MODEL	LORA RANK	EM (%)
LLAMA 3.2 1B	32	15.86
LLAMA 3.2 1B	256	13.28
LLAMA 3.2 3B	32	37.20
LLAMA 3.2 3B	256	25.52
QWEN 2 0.5B	32	6.96
QWEN 2 0.5B	256	6.72

5.2. Comparisons

Six primary experiments are conducted:

1. **Direct Baseline (DB)** (Baseline-1) : Fine-tune on the entire dataset in a single pass (rank=32).
2. **Sequential No-Replay (SNR)** (Baseline-2) : Step through levels but discard earlier-level data each time (rank=32).
3. **SNR + PLRS** (Baseline-3) : SNR with reducing rank at each stage($256 \rightarrow 128 \rightarrow 64 \rightarrow 32 \rightarrow 16$).
4. **Sequential Full Replay (SFR) + Fixed Rank** (Baseline-4) : Replay all older tasks at each stage, but keep rank=32 throughout.
5. **SFR + PLRS** (Baseline-5) : This approach includes re-exposing older tasks and reducing rank at every level ($256 \rightarrow 128 \rightarrow 64 \rightarrow 32 \rightarrow 16$).
6. **SFR + PLRS + No Final Shrink** (Baseline-6): Our final approach, re-exposing older tasks and reducing rank ($256 \rightarrow 128 \rightarrow 64 \rightarrow 32 \rightarrow 32$) but not shrinking at level 5.

5.3. Results and Analysis

Key takeaways at a glance :

- **Synergy, not additivity.** Replay *or* PLRS in isolation yields at best baseline-level EM (14–15%); the *combination* pushes accuracy to **20.32%**, a gain of 4.46% over the direct Baseline 1.
- **Parameter efficiency.** A single-pass, rank-256 baseline *underperforms* rank-32 SFT on all models (e.g., 13.28% vs. 15.86% on LLaMA-1B), showing that *capacity scheduling*, not just rank budget, drives the improvement.
- **Positive backward transfer.** Our final recipe boosts Level-1 accuracy from 42.1% (Baseline 1) to **53.3%** (Baseline 6), even after the model has mastered harder levels. Rehearsal + isolation not only preserves knowledge it actively refines earlier skills.
- **Cross-backbone robustness.** Gains of 2–7% hold across four model architectures from 0.5B to 3B, suggesting that the curriculum–replay–PLRS triad is architecture-agnostic.

Table 3 shows final ablations for LLaMA 3.2 1B. Notably, replay alone (SFR + fixed rank) can *drop* performance to 12.38%, even below direct Baseline 1 (15.86%). Combining replay with rank shrinking yields 17.22%, and one final tweak no rank shrink at Level-5 raises accuracy to 20.32%. Figure 5 illustrates this improvement with a representative math problem.

The final approach (Baseline 6) is extended to LLaMA 3.2 3B, Qwen2.5Math (1.5B), and Qwen 2 (0.5B). It could be seen that improvements over the respective direct baselines (Baseline 1) are consistent: 2–7% absolute, as shown in Figure 2. Specifically, LLaMA 3.2 1B improves by +4.46%, LLaMA 3.2 3B by +6.92%, Qwen2.5Math 1.5B by +4.10%, and Qwen 2 0.5B by +1.92%.

Table 3. Ablations on LLaMA 3.2 1B.

TECHNIQUE	EM (%)	REPLAY	SHRINK
DB (ALL-AT-ONCE)	15.86	NO	NO
SNR + FIXED RANK	14.28	NO	NO
SNR + PLRS	15.10	NO	YES
SFR + FIXED RANK	12.38	YES	NO
SFR + PLRS	17.22	YES	YES
+NO FINAL SHRINK	**20.32**	YES	YES

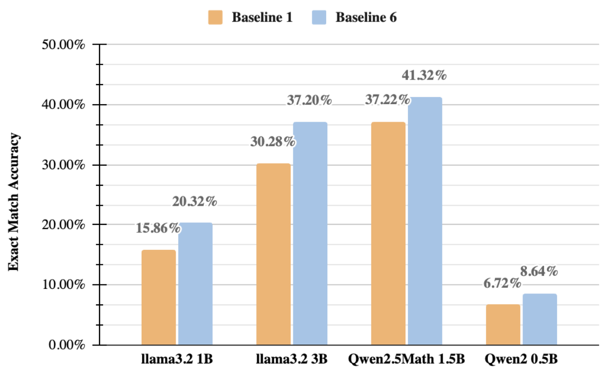

Figure 2. EM improvement on Baseline 6 compared to Baseline 1.

5.4. Interpretation of Sequential Strategies

(a) Sequential, No Replay. Training each level in isolation triggers *catastrophic forgetting*: Level-1 accuracy sinks from 42.1% (Baseline 1) to 34.1% by the time the model reaches Level-5, and overall EM drops by 1.6% compared to direct SFT.

(b) Sequential + Full Replay. Rehearsal revives easy-tier performance (Level-1 rises to 39.4%, a gain of +3.3%), but overwhelms the adapter: overall EM crashes to 12.4% (a 3.5% drop from Baseline 1) and Level-5 drops to just 2.3%. *Lesson: replay without extra capacity merely overfits easy patterns.*

(c) Replay + PLRS. The combination of rehearsal with rank shrinking yields a balanced lift: Level-1 improves to 45.54% (3%), Level-5 reaches 4.31%, and overall EM rises to 17.2% (improvement over Baseline 1).

(d) Replay + PLRS without the last shrink . Retaining r=32 at Stage-5 injects fresh capacity for the hardest level, nudging overall EM to **20.3%** a +4.5% gain over direct SFT (Baseline 1) while preserving lower-level improvements from earlier stages. This configuration defines our **final recipe** (Baseline 6).

Take-home pattern. *Capacity isolation* (PLRS) protects against forgetting, *replay* refines early skills, and both are required to unlock Level-5 accuracy without sacrificing Level-1 performance. Refer Table 4 for details.

Error–trajectory analysis: *what* the extra accuracy is made of. Exact-match scores hide *which* problems are gained, lost, or re-learned across stages. We therefore assign every test item to one of six mutually exclusive buckets (Table 5):

- **stable_correct**: solved at every stage after first exposure
- **lost**: solved at own level but wrong in the final model
- **recovered**: solved, forgotten, then solved again
- **newly_solved**: first solved at a later stage
- **never_solved**: always wrong
- **other**: mixed, non-terminal flips

Labels are computed by scanning the 0/1-correctness vector from the problem's introduction level through Stage 5

Following patterns stand out : (refer Table 5)

1. **Replay alone amplifies forgetting.** It introduces 332 *newly_solved* items, but inflates the *lost* bucket to **459** (+7 % vs. SNR) and shrinks the *stable_correct* set to **250**. Easy samples dominate the gradient budget, erasing mid-level skills.
2. **PLRS alone protects but discovers little.** Shrinking rank cuts *lost* to 410 and restores *stable_correct* (332) to the no-replay baseline, yet adds only 25 extra *newly_solved* items. Capacity isolation without rehearsal is safe but conservative.
3. **Replay *plus* PLRS is decisively synergistic.** The combined recipe slashes *lost* to just **176** *and* more than doubles *stable_correct* to **569**. It also delivers the largest haul of *newly_solved* problems (408) and the most *recovered* cases (39).

Table 4. Level-wise EM accuracy (%) after the final stage (Level 5) of Sequential Fine-Tuning.

BASELINE	LEVEL 1	LEVEL 2	LEVEL 3	LEVEL 4	LEVEL 5	OVERALL
DIRECT - BASELINE 1	42.11	25.50	18.48	9.72	4.08	15.86
SNR - BASELINE 2	34.10↓	24.16↓	16.62↓	8.57↓	4.31↑	14.28
SNR + PLRS - BASELINE 3	38.44↓	23.94↓	18.04↓	9.47↓	4.08	15.10
SFR - BASELINE 4	39.36↓	21.03↓	13.35↓	6.43↓	2.27↓	12.38
SFR + PLRS - BASELINE 5	45.54↑	28.19↑	19.01↑	11.37↑	4.31↑	17.22
SFR + PLRS + NO FINAL SHRINK - BASELINE 6	53.32↑	34.45↑	21.13↑	14.00↑	4.98↑	20.32

Note: ↑ indicates improvement over the Direct baseline; ↓ indicates performance drop.
In **SNR**, only Level 5 shows improvement, suggesting earlier capabilities are overwritten during sequential fine-tuning.

Table 5. Problem categories by training recipe (darker = more extreme; orange = worse, blue = better).

CATEGORY	SNR	SFR	SNR + PLRS	SFR + PLRS	NO FINAL SHRINK
LOST	429	459	410	176	365
NEVER_SOLVED	3778	3845	3756	3776	3704
OTHER	79	77	79	32	70
NEWLY_SOLVED	361	332	386	408	451
RECOVERED	29	37	37	39	49
STABLE_CORRECT	324	250	332	569	361

4. **Final-stage rank restores head-room (Baseline 6).** Retaining a full $r = 32$ adapter at Level 5 injects fresh capacity for the hardest questions and pushes *newly_solved* to an all-time high of **451** (+43 vs. SFR + PLRS). It also lifts the *recovered* bucket to **49**, showing extra hard-problem wins can cascade backwards. The cost is a partial rebound in *lost* items (365 vs. 176), indicating that some early-stage parameters are overwritten when the final, larger slice is trainable. In short, *extra rank at the last stage buys exploration power but reopens a modest forgetting value*; practitioners can tune this trade off by choosing whether or by how much to shrink on the final curriculum step.

Why this matters. The bucket view shows that the EM gain of Replay + PLRS is not merely "finding a few more answers"; it reflects a simultaneous **preservation** of prior knowledge and a **net expansion** into previously incorrect territory a strong indicator of genuine curriculum learning rather than over-fitting.

Back-ward transfer, not just retention. The model didn't merely keep its level-1 skills it improved them after seeing harder material. "The best variant (Replay + PLRS, no final shrink) raises level-1 accuracy to 53%, a +11% gain over the basline - 1. This indicates positive backward transfer: after mastering harder problems the model not only preserves but improves its performance on easier tasks. The effect arises from repeated exposure (replay) plus parameter isolation (rank-shrinking), which together prevent forgetting while allowing new competencies to accumulate.

Reconciling the replay paradox. Full replay seems to cut two ways: it **raises** Level-1 EM (36.4 % → 39.4 %) yet **inflates** the *lost* bucket to 459 items (Table 5).

PLRS breaks this trade-off by assigning each stage its own shrinking slice, letting replayed gradients reinforce earlier skills without overwriting later ones preserving Level-1 gains without inflating the *lost* bucket.

1. **Easy-task dominance.** Thousands of Level-1 examples monopolise gradients, so the rank-32 adapter allocates extra parameter capacity to them hence the Level-1 bump.

2. **Capacity overwrite.** The same fixed adapter must also encode mid-/high-level reasoning. Gradients from easy tasks overwrite weights that formerly solved harder items, which then migrate to the *lost* category.

6. Limitations and Future Work

While the proposed PLRS framework delivers strong performance gains with minimal parameter overhead, it also presents several limitations and open research questions.

First, the current rank-shrinking schedule was manually designed; although effective, it may be suboptimal. Future work could focus on learning adaptive rank schedules that dynamically respond to training signals such as gradient norms or task difficulty. Similarly, our approach relies on full-history replay, which is memory-intensive. A selective or uncertainty-based replay strategy could maintain performance while reducing compute demands.

Moreover, PLRS benefits from the MATH dataset's naturally tiered structure. Applying this method to less-structured domains may require more sophisticated curriculum design or automatic task partitioning. Finally, despite LoRA's efficiency, multi-stage fine-tuning still incurs non-trivial compute costs—especially on larger backbones—underscoring the need for scalable optimization strategies.

Promising Future Directions.

- **Adaptive replay scheduling.** Instead of naively replaying all prior data, future work could develop policies that adapt replay ratios per stage or level based on metrics like validation loss, uncertainty estimates, or gradient conflict.
- **Dynamic rank allocation.** Replace the fixed PLRS schedule with a learnable controller that allocates LoRA rank on-the-fly.

7. Conclusion

We propose a capacity-aware sequential learning framework for small language models that integrates three key components: (**i**) level-wise training from easy to hard, (**ii**) rehearsal of earlier problems to mitigate forgetting, and (**iii**) a novel *Progressive LoRA Rank Shrinking* (PLRS) strategy that isolates stage-specific competencies via structured parameter allocation.

When applied to the MATH benchmark, our approach yields consistent exact-match improvements of **+2–7%** across four open source models ranging from 0.5B to 3B parameters all within a 4-bit quantized memory footprint.

Beyond raw accuracy, our error trajectory analysis reveals that the combination of replay and PLRS does more than preserve prior skills it actively *enhances* them, demonstrating **positive backward transfer**.

These findings underscore a broader insight: with careful orchestration of task progression and parameter allocation, even lightweight models can exhibit emergent reasoning capabilities challenging the assumption that scale alone is the path to skill.

Impact Statement

"This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here."

References

Biderman, D., Portes, J., Ortiz, J. J. G., Paul, M., Greengard, P., Jennings, C., King, D., Havens, S., Chiley, V., Frankle, J., Blakeney, C., and Cunningham, J. P. Lora learns less and forgets less, 2024. URL https://arxiv.org/abs/2405.09673.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Das, D., Banerjee, D., Aditya, S., and Kulkarni, A. MATHSENSEI: A tool-augmented large language model for mathematical reasoning. In Duh, K., Gomez, H., and Bethard, S. (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 942–966, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.54. URL https://aclanthology.org/2024.naacl-long.54/.

Dong, G., Yuan, H., Lu, K., Li, C., Xue, M., Liu, D., Wang, W., Yuan, Z., Zhou, C., and Zhou, J. How abilities in large language models are affected by supervised fine-tuning data composition. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 177–198, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.12. URL https://aclanthology.org/2024.acl-long.12/.

Gou, Z., Shao, Z., Gong, Y., yelong shen, Yang, Y., Huang, M., Duan, N., and Chen, W. ToRA: A tool-integrated reasoning agent for mathematical problem solving. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Ep0TtjVoap.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. In Vanschoren, J. and Yeung, S. (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/

be83ab3ecd0db773eb2dc1b0a17836a1-Paper-round2.pdf.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Huang, J., Cui, L., Wang, A., Yang, C., Liao, X., Song, L., Yao, J., and Su, J. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. *arXiv preprint arXiv:2403.01244*, 2024.

Li, C., Yuan, Z., Yuan, H., Dong, G., Lu, K., Wu, J., Tan, C., Wang, X., and Zhou, C. MuggleMath: Assessing the impact of query and response augmentation on math reasoning. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10230–10258, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.551. URL https://aclanthology.org/2024.acl-long.551/.

Longpre, S., Hou, L., Vu, T., Webson, A., Chung, H. W., Tay, Y., Zhou, D., Le, Q. V., Zoph, B., Wei, J., et al. The flan collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning*, pp. 22631–22648. PMLR, 2023.

Lu, Z., Zhou, A., Ren, H., Wang, K., Shi, W., Pan, J., Zhan, M., and Li, H. MathGenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of LLMs. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2732–2747, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.151. URL https://aclanthology.org/2024.acl-long.151/.

Tang, Z., Zhang, X., Wang, B., and Wei, F. Mathscale: Scaling instruction tuning for mathematical reasoning. *arXiv preprint arXiv:2403.02884*, 2024.

Wang, Y., Mishra, S., Alipoormolabashi, P., Kordi, Y., Mirzaei, A., Arunkumar, A., Ashok, A., Dhanasekaran, A. S., Naik, A., Stap, D., et al. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*, 2022.

Wang, Y., Liu, Y., Shi, C., Li, H., Chen, C., Lu, H., and Yang, Y. InsCL: A data-efficient continual learning paradigm for fine-tuning large language models with instructions. In Duh, K., Gomez, H., and Bethard, S. (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 663–677, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.37. URL https://aclanthology.org/2024.naacl-long.37/.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

You, W., Yin, S., Zhao, X., Ji, Z., Zhong, G., and Bai, J. MuMath: Multi-perspective data augmentation for mathematical reasoning in large language models. In Duh, K., Gomez, H., and Bethard, S. (eds.), *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 2932–2958, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.185. URL https://aclanthology.org/2024.findings-naacl.185.

Yue, X., Qu, X., Zhang, G., Fu, Y., Huang, W., Sun, H., Su, Y., and Chen, W. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*, 2023.

A. Appendix

Qwen Prompt

```
<|im_start|> system
You are an expert math assistant. <|im_end|>
<|im_start|> user
Solve the following math problem: {problem}. Show all steps and the final answer in LaTeX format in a box like \boxed{}.
<|im_start|> assistant {answer}
```

Figure 3. Qwen Prompt Format

Llama Prompt

```
<|begin_of_text|>
<|start_header_id|>system <|end_header_id|>
You are an expert math assistant<|eot_id|>
<|start_header_id|>user <|end_header_id|>
Solve the following math problem: {problem} Show all steps and include the final answer in LaTeX format in a box like \boxed{}.<|eot_id|>
<|start_header_id|> assistant <|end_header_id|> {answer}
```

Figure 4. Llama Prompt Format

Algebra Problem Example

Question: If $(x, y) = (3, 9)$, what is $y^2 - 3xy + 8$?
Level: 1
Subject: Algebra
Ground Truth: We have

$$y^2 - 3xy + 8 = 9^2 - 3(3)(9) + 8 = 81 - 81 + 8 = \boxed{8}.$$

Baseline 1 Solution:
The expression $y^2 - 3xy + 8$ is the expansion of $(y - 3x)^2$.
Substituting $(3, 9)$ for (x, y):

$$(9 - 3 \cdot 3)^2 = 9^2 - 3 \cdot 9 \cdot 3 + 3^2 = 81 - 81 + 9 = \boxed{9}.$$

Baseline 6 Solution:
Step 1: Substitute the given values of x and y into the equation.
Step 2:Calculate y^2: $9^2 = 81$.
Step 3:Calculate $3xy$: $3 \times 3 \times 9 = 81$.
Step 4: Substitute values into the equation: $81 - 81 + 8$.
Step 5:Simplify: $81 - 81 + 8 = 8$.
Final Answer: $\boxed{8}$.

Figure 5. Comparison of Ground Truth, Baseline 1, and Baseline 6 Solutions (instance where Baseline 6 is correct and Baseline 1 is incorrect)

The project code and supplementary materials are available at: https://gadmin7.github.io/forget_less_solve_more/

Table 6. Training arguments for LLaMA 3.2 1B baselines.

SETUP BASELINE 6

PARAMETER	VALUE
BATCH SIZE	4
LEARNING RATE	5E-5
NUMBER OF EPOCHS (PER LEVEL)	5
PRECISION	FP16
OPTIMIZER	PAGED_ADAMW_8BIT
WARMUP RATIO	0.1
EVALUATION STRATEGY	STEPS
LOAD BEST MODEL AT END	TRUE
METRIC FOR BEST MODEL	EVAL_LOSS
LORA RANKS	$256 \rightarrow 128 \rightarrow 64 \rightarrow 32 \rightarrow 32$
LORA ALPHA	$512 \rightarrow 256 \rightarrow 128 \rightarrow 64 \rightarrow 64$
LORA DROPOUT	0.2
QUANTIZATION	4-BIT NF4

Table 7. Training arguments for LLaMA 3.2 1B baselines.

SETUP BASELINE 1

PARAMETER	VALUE
BATCH SIZE	4
LEARNING RATE	5E-5
NUMBER OF EPOCHS	5
PRECISION	FP16
OPTIMIZER	PAGED_ADAMW_8BIT
WARMUP RATIO	0.1
EVALUATION STRATEGY	STEPS
LOAD BEST MODEL AT END	TRUE
METRIC FOR BEST MODEL	EVAL_LOSS
LORA RANK	32 (FIXED)
LORA ALPHA	64
LORA DROPOUT	0.2
QUANTIZATION	4-BIT NF4

Table 8. EM accuracy comparison of Baseline 1 and Baseline 6 across models.

LEVEL	TECHNIQUE	LLAMA3.2 1B	LLAMA3.2 3B	QWEN2.5MATH 1.5B	QWEN2 0.5B
L1	BASELINE 1	42.10	64.53	69.79	19.22
	BASELINE 6	53.31	72.54	83.30	27.92
	IMPROVEMENT	11.21	8.01	13.50	8.70
L2	BASELINE 1	25.50	45.86	54.59	10.07
	BASELINE 6	34.45	55.37	61.86	15.32
	IMPROVEMENT	8.95	9.51	7.27	5.26
L3	BASELINE 1	18.47	36.43	43.06	6.81
	BASELINE 6	21.13	45.62	48.81	8.84
	IMPROVEMENT	2.66	9.20	5.75	2.03
L4	BASELINE 1	9.71	22.82	30.40	3.79
	BASELINE 6	14.00	28.75	33.03	3.62
	IMPROVEMENT	4.29	5.93	2.64	-0.16
L5	BASELINE 1	4.07	10.05	16.01	2.95
	BASELINE 6	4.98	13.82	14.80	2.19
	IMPROVEMENT	0.91	3.78	-1.21	-0.76

SUBJECT	TECHNIQUE	LLAMA3.2 1B	LLAMA3.2 3B	QWEN2.5MATH1.5B	QWEN2 0.5B
ALGEBRA	BASELINE 1	24.17	44.65	54.76	8.59
	BASELINE 6	31.76	55.86	59.39	11.88
	IMPROVEMENT	7.59	11.20	4.63	3.29
COUNTING	BASELINE 1	13.71	27.64	28.69	5.91
	BASELINE 6	15.82	35.23	37.34	6.96
	IMPROVEMENT	2.11	7.60	8.65	1.05
GEOMETRY	BASELINE 1	12.96	23.80	29.44	7.10
	BASELINE 6	16.07	30.48	35.91	6.89
	IMPROVEMENT	3.11	6.68	6.47	-0.21
INT. ALGEBRA	BASELINE 1	5.98	13.73	20.16	3.65
	BASELINE 6	5.75	14.62	17.72	3.77
	IMPROVEMENT	-0.23	0.89	-2.44	0.11
NUMBER THEORY	BASELINE 1	12.96	23.33	30.56	5.56
	BASELINE 6	13.14	27.41	32.04	4.63
	IMPROVEMENT	0.18	4.07	1.48	-0.93
PREALGEBRA	BASELINE 1	24.56	45.92	52.70	9.99
	BASELINE 6	36.73	58.55	65.21	15.84
	IMPROVEMENT	12.17	12.63	12.51	5.86
PRECALCULUS	BASELINE 1	7.50	16.30	23.44	4.03
	BASELINE 6	8.05	17.22	20.33	5.13
	IMPROVEMENT	0.55	0.92	-3.11	1.10