SELF-TRAINING ON UNPROMPTED SAMPLES CAN RAISE UTILITY AND LOWER VERBATIM EXTRACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Can a language model improve by training on its own unprompted text, without an external teacher, verifier, or reward model? How does such training affect verbatim extraction of its pretraining data? We study these questions by fine-tuning four base language models from three families on fixed corpora they generate from a minimal prefix, with no task specification or external supervision. We report three findings. First, self-generated data can improve reasoning, math, and code benchmarks beyond what a matched generic corpus provides, although the gains depend on the model, the task, and the evaluation protocol. Second, the same training can substantially reduce measured verbatim extraction of documented pretraining data without a forget set, privacy objective, or targeted unlearning, and at some points in training lower extraction coincides with higher mean benchmark scores, even when individual benchmarks decline. Third, transfer depends on the source: for a fixed student, its own samples help more than samples from larger or differently trained models on most benchmarks, and corpora with similar mean likelihood under the student can have different downstream effects. These findings are consistent with *latent capability resurfacing*: self-training may amplify capabilities already present in the student when the synthetic corpus is compatible with it.

1 INTRODUCTION

Training a language model on its own outputs is supposed to fail. Recursive self-training can produce distributional drift and eventual collapse (Shumailov et al., 2024; Alemohammad et al., 2024a), and modern post-training pipelines typically treat synthetic data as unreliable unless it is filtered through verification, reward models, or correctness checks (Feng et al., 2025; Zelikman et al., 2022; Singh et al., 2024). Yet a growing body of work complicates this picture. Zhang et al. (2026) improve code generation by fine-tuning on raw, unverified model outputs without filtering them. Reward-free self-training on reasoning responses yields gains without external verifiers (Li et al., 2025). Karan & Du (2026) recover much of the reasoning-benchmark benefit associated with reinforcement learning through inference-time sampling alone. In vision, unverified self-generated samples can also improve generative models beyond their training distribution (Alemohammad et al., 2024b; 2026). Across these disparate settings, a pattern emerges: weak, unstructured training signals with no task-specific supervision and no explicit teacher can still help. Something already present in the pretrained model is being accessed or amplified.

Several of these works propose, independently, the same candidate explanation: weak self-training resurfaces latent capabilities already encoded during pretraining and does not inject new knowledge. The evidence, however, is scattered across prompt-conditioned code, reward-style reasoning, and inference-only sampling, and each setting introduces its own confounds. What has been missing is a controlled setting in which task prompts, external reward, and explicit teacher annotations are absent by construction, so that any observed gain must arise from the interaction between a pretrained student and a weak self-generated signal.

We also ask how self-training affects verbatim extraction of pretraining data. Language models can reproduce training sequences verbatim, and extraction rates increase with model size and data duplication (Carlini et al., 2021; 2023). Targeted unlearning methods specify the content to remove,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

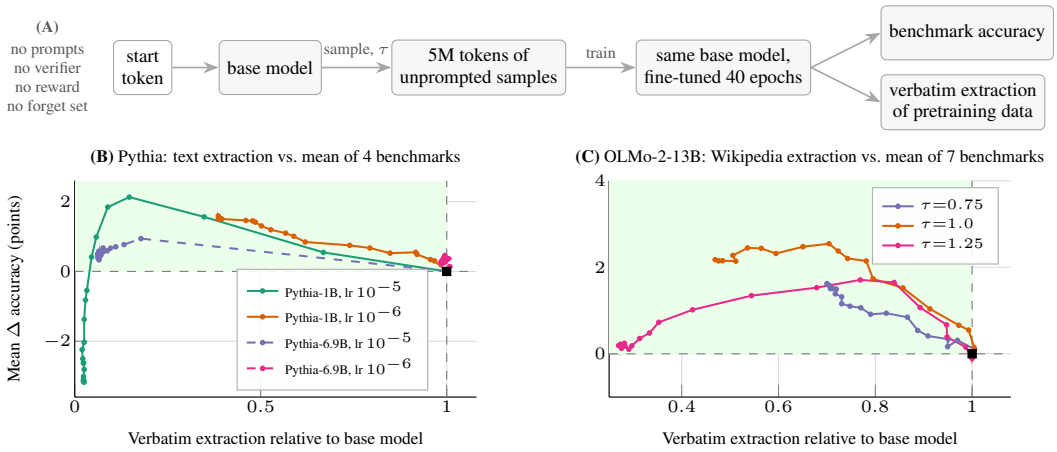


Figure 1: **Training a base model on its own unprompted samples produces checkpoints with a higher mean benchmark score and lower verbatim extraction than the base model. (A)** The pipeline: a base model samples text from a fixed minimal prefix, and the same model is fine-tuned on those samples; 5M tokens is a nominal budget (Section 3). **(B, C)** Each point shows results at one evaluated epoch, averaged over three training sets. We evaluate every two epochs through epoch 40. Lines connect points in training order, beginning with the base model (square). The shaded region marks higher mean benchmark scores and lower extraction than base; individual benchmarks can still decline there.

often at a cost to utility (Jang et al., 2023; Maini et al., 2024). If weak self-training amplifies distributed structure already present in the model, it may also weaken the sequence-specific recall that competes with that structure.

We therefore fine-tune base language models on plain text generated from a fixed minimal prefix. Generation uses no task prompts or specifications, and the training data receives no quality or correctness filtering or external supervision. The base model generates the corpus once, and we reuse it throughout fine-tuning, so the procedure consists of one round of self-training. The model itself generates any reasoning traces or factual content in the corpus; we supply no task examples during generation. If latent capability resurfacing is real, this is where it should appear in its cleanest form. Our *latent capability resurfacing* hypothesis has three parts. First, weak self-training can make pretrained capabilities more apparent in the model's behavior. Second, in favorable regimes, it may strengthen distributed task structure at the expense of sequence-specific recall. Third, its effect depends on the compatibility between the synthetic corpus and the student, a property of the source-student pair and not of the data alone. We investigate this hypothesis through changes in benchmark scores and verbatim extraction, while leaving the underlying mechanism unresolved.

We report three main findings. First, self-generated text improves Qwen2.5-0.5B scores on GSM8K, Minerva-MATH, and HumanEval in ways the Common Corpus baseline does not reproduce. The benchmarks that improve differ across model families, consistent with a role for capabilities supported by each pretrained model. Second, self-training reduces measured verbatim extraction on Pythia and OLMo-2-13B, whose pretraining data is documented. At some matched training epochs, lower mean extraction coincides with higher mean benchmark scores (Figure 1B and C). Third, transfer depends on the source corpus: self-generated data yields the largest gains on most benchmarks, while corpora with similar mean negative log-likelihood (NLL) under the student can have different downstream effects.

2 Related Work

Studies of model collapse mainly examine repeated rounds of training on the previous model's outputs (Shumailov et al., 2024; Alemohammad et al., 2024a). Verified self-training instead retains generations that pass a correctness check (Zelikman et al., 2022; Singh et al., 2024; Feng et al., 2025). We study one round of fine-tuning on a fixed corpus of unverified samples from a pretrained base model.

The closest prior work is Zhang et al. (2026), who improve code generation by fine-tuning on unverified solutions to competitive-programming prompts, without a verifier or stronger teacher. Related results appear in reward-free self-training for reasoning (Li et al., 2025) and self-training for generative vision models (Alemohammad et al., 2024b; 2026).

A parallel line argues that these gains come from capabilities already latent in the base model. Shao et al. (2025) show that random, incorrect, or format-only rewards in reinforcement learning with verifiable rewards (RLVR) still produce large gains on Qwen2.5-Math but not on other model families. Karan & Du (2026) reach a similar conclusion from inference alone, showing that training-free iterative sampling nearly matches reinforcement-learned models, and Ji et al. (2026) obtain related gains through inference-time distribution sharpening. Tan et al. (2026) argue that RLVR on the conditional distribution $p(y \mid x)$ is bounded by the base model's existing output distribution. They propose reward-driven updates to the marginal $p(y)$, removing the question during the update while retaining prompt-conditioned rollouts. Our generation uses no task prompts, and our training uses no rewards.

Language models also memorize and reproduce verbatim training data (Carlini et al., 2021), with extraction rates that grow with model size and duplication (Carlini et al., 2023). Criteria beyond greedy exact match, such as near-verbatim matching (Cooper et al., 2026) and probabilistic extraction (Hayes et al., 2025), find more memorization. Memorized examples can also become less extractable as training continues on other data (Jagielski et al., 2023). Targeted unlearning methods specify what to remove, using approaches such as gradient ascent (Jang et al., 2023) or preference optimization on forget sets (Maini et al., 2024), often at a cost to utility. Scholten et al. (2026) instead train on the model's own generations to trigger collapse for specified forget queries while preserving utility. Our setting uses neither task-conditioned generation nor a specified forget set.

3 Setup

Models. All students are pretrained base language models. We use Qwen2.5-0.5B (Yang et al., 2024), whose pretraining data is not public, to evaluate benchmark utility and compare synthetic sources. We evaluate both utility and extraction on Pythia-1B and Pythia-6.9B (Biderman et al., 2023), trained on the Pile (Gao et al., 2020), and OLMo-2-1124-13B (Team OLMo et al., 2025), whose stage-1 training mix, OLMo-Mix-1124, is public. We call the model being fine-tuned the *student* and the model generating its training data the *source*. Except in Section 6, they are the same model.

Synthetic corpora and training. We generate each corpus from a fixed minimal prefix s: a newline for Qwen2.5-0.5B, which has no beginning-of-sequence token, and <|endoftext|> for Pythia and OLMo 2. Generation uses no task prompts, instructions, or specifications. We sample at temperature τ without top-k or top-p truncation, enforcing a minimum of 50 tokens and stopping at an end-of-generation token or 1,024 tokens. We apply 8-gram decontamination against the training, validation, and test splits of every evaluation benchmark, without filtering samples for quality or correctness. Each corpus is generated once from a fixed source checkpoint and reused for all 40 epochs. From each decontaminated corpus, we draw training sets $\mathcal{D}$ with a nominal budget of 5M tokens. For Qwen2.5-0.5B, we estimate this budget at four characters per token, so actual tokenizer-measured counts can differ across corpora. Starting from the base parameters θ_0, we fine-tune all student parameters for 40 epochs with AdamW on the next-token loss

$$\mathcal{L}(\theta) \;=\; -\sum_{x\in\mathcal{D}}\sum_{t}\log p_\theta(x_t \mid s, x_{<t})\,, \qquad x \sim p^{\tau}_{\mathrm{src}}(\cdot \mid s), \tag{1}$$

where p^{τ}_{src} denotes sampling from the source at temperature τ, followed by the length limits and decontamination above. Within each model and learning rate, only the training corpus varies. Each model's paragraph in Section 4 gives its temperatures, learning rate, batch size, and number of training sets, and Appendix D lists the key training and evaluation settings.

Evaluation. We evaluate Qwen2.5-0.5B and OLMo-2-13B zero-shot with lm-evaluation-harness (Gao et al., 2024) on MMLU (Hendrycks et al., 2021), ARC-Challenge (Clark et al., 2018), GSM8K (Cobbe et al., 2021), HellaSwag (Zellers et al., 2019), TruthfulQA (Lin et al., 2022), Minerva-MATH (Lewkowycz et al., 2022), and HumanEval (Chen et al., 2021). GSM8K is scored

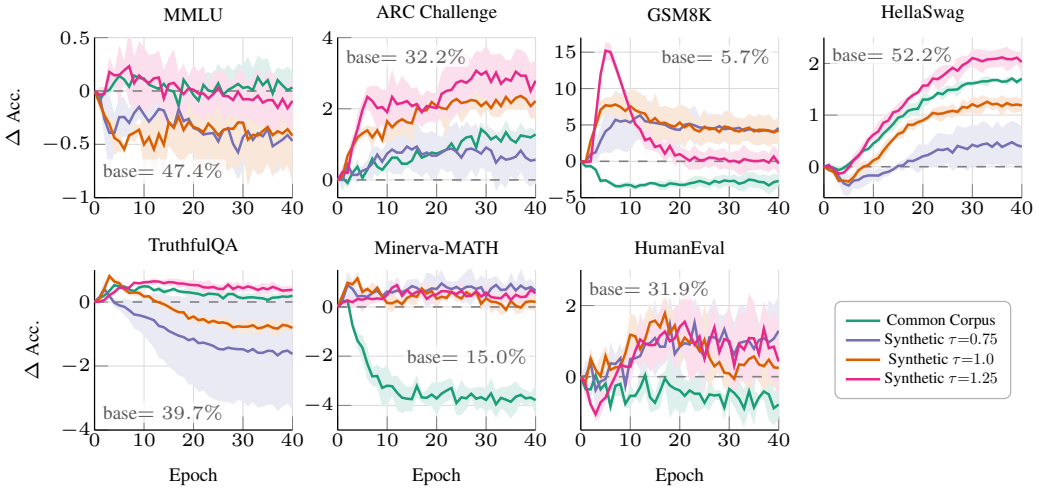


Figure 2: **Training Qwen2.5-0.5B on its own unprompted samples raises GSM8K, HumanEval, and Minerva-MATH accuracy; training on Common Corpus does not reproduce these gains.** Δ accuracy relative to the base model over 40 epochs of training on the model's own samples at three temperatures or on Common Corpus, each with the nominal budget of Section 3. Shaded bands show ±1 std. across three subsets.

with flexible-extract, Minerva-MATH with `math_verify`, and HumanEval as pass@1 estimated from 10 samples per problem at temperature 0.2. For Qwen2.5-0.5B we also apply standard few-shot protocols (ARC-Challenge 25-shot, GSM8K 8-shot, HellaSwag 5-shot, Minerva-MATH 4-shot). Pythia is evaluated on ARC-Challenge, HellaSwag, ARC-Easy, and WinoGrande (Sakaguchi et al., 2020). *Mean accuracy change* is the unweighted mean change in benchmark accuracy relative to the base model, in percentage points.

Verbatim extraction. We measure extractability with the prefix-extraction attack of Carlini et al. (2021): we prompt the model with the first 50 tokens of a pretraining sequence, decode 50 tokens greedily, and count the sequence as extracted if the output matches the true continuation exactly. For OLMo-2-13B, we also use two alternative criteria. Near-verbatim extraction counts greedy outputs within five token edits of the true continuation, motivated by Cooper et al. (2026). Probabilistic extraction counts sequences whose true continuation has probability at least 10^{-3} under top-40 sampling (Hayes et al., 2025).

Reporting. Figures show the evaluated checkpoints: every epoch for Qwen2.5-0.5B and every two epochs for Pythia and OLMo-2-13B. Curves show the mean over three training sets, with shaded bands of ± 1 standard deviation. Unless we name a checkpoint, each reported utility gain is the maximum of that benchmark's mean curve across evaluated checkpoints. Extraction reductions are relative changes in extracted-sequence counts at epoch 40, when extraction is at or near its minimum. We use *configuration* for a model with a given learning rate or temperature, and *replicate* for each of its three training sets.

4 Self-generated data produces transient, model-dependent gains

Qwen2.5-0.5B. Qwen2.5-0.5B samples from a newline token at $\tau \in \{0.75, 1.0, 1.25\}$. We train on three sets per temperature, using a learning rate of 10^{-6}, an effective batch size of 64, and sequences packed to 2,048 tokens. For the generic baseline, we sample the publicly available multilingual Common Corpus (Langlais et al., 2026) using the same nominal token budget.

Figure 2 shows different training trajectories across tasks. On HumanEval, all three sampling temperatures produce modest gains of up to 1.8 points, with no clear temperature ordering. Minerva-MATH improves by up to 1.2 points. On GSM8K, the gain at τ=1.25 peaks early at 15.2 points and then falls toward baseline. The gains at τ=0.75 and τ=1.0 are smaller but more stable. Common Corpus does not reproduce these gains under the same training procedure: it improves none of GSM8K, Minerva-MATH, or HumanEval by more than 0.3 points, and performance on the latter two

tasks declines with training. On ARC-Challenge and HellaSwag, its gains fall within the range of the synthetic conditions. Both corpus choice and sampling temperature affect the size and persistence of the improvements.

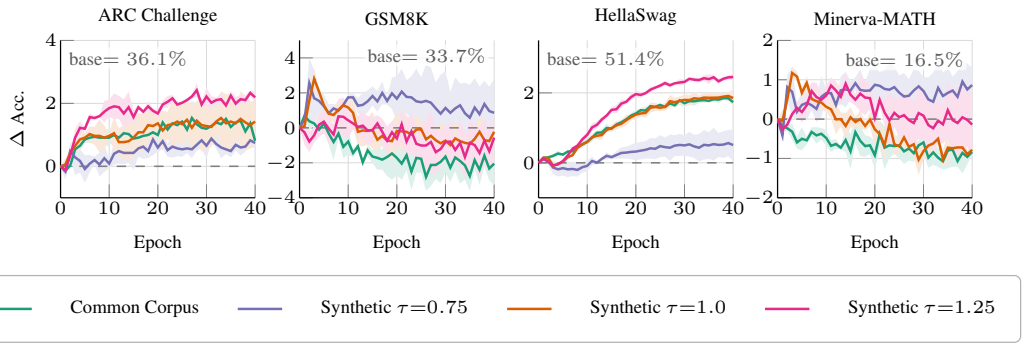


Figure 3: **Under few-shot evaluation, ARC-Challenge and HellaSwag retain gains, while the maximum GSM8K gain is below 3 points.** Qwen2.5-0.5B Δ accuracy under ARC-Challenge 25-shot, GSM8K 8-shot, HellaSwag 5-shot, and Minerva-MATH 4-shot evaluation over 40 epochs. Shaded bands show ±1 std. across subsets.

Under few-shot evaluation (Figure 3), the best synthetic temperature outperforms Common Corpus on ARC-Challenge and HellaSwag. The maximum GSM8K gain is much smaller, at 2.8 points, and the gains shrink with further training. This sensitivity to evaluation context suggests that part of the zero-shot gain may reflect how the model responds to the prompt. On Minerva-MATH, τ=0.75 sustains a positive mean gain later in training, whereas the other temperatures and Common Corpus finish below base.

OLMo-2-13B. OLMo-2-13B shows a different pattern. It samples from its <|endoftext|> token at the same three temperatures, and we train on three sets per temperature with the same learning rate, effective batch size, sequence length, and number of epochs as Qwen2.5-0.5B (Figure 4). GSM8K and TruthfulQA improve by up to 23.4 and 3.0 points, respectively, and remain near their peaks through epoch 40. Under 8-shot evaluation of one training set, GSM8K instead falls below base after early training (Appendix B). Minerva-MATH and HumanEval improve early in some settings before declining, while ARC-Challenge and MMLU remain near base initially and finish below it.

Pythia. Pythia-1B and Pythia-6.9B sample from <|endoftext|> at τ=1.25, and we train three sets per configuration with learning rates 10^{-5} and 10^{-6}, effective batch size 32, and 1,024-token sequences (Figure 5). At 10^{-5}, Pythia-6.9B retains gains of about 2 points on ARC-Challenge and HellaSwag through epoch 40, while ARC-Easy and WinoGrande decline. Pythia-1B at 10^{-6} shows modest gains on all four benchmarks. Pythia-1B at 10^{-5} improves early and then declines, whereas Pythia-6.9B at 10^{-6} changes little.

Both the tasks that improve and the duration of the gains differ across models. Qwen2.5-0.5B and OLMo-2-13B show different trajectories on the same seven benchmarks, and Pythia improves on the comprehension benchmarks we evaluate. The differences are consistent with a role for pretrained capabilities, although the experiments do not isolate pretraining history from other model differences.

5 Self-Training Reduces Measured Verbatim Extraction

The resurfacing hypothesis suggests that favorable self-training may reduce verbatim extraction as task performance improves. We examine this prediction in Pythia and OLMo-2-13B, whose pretraining data is documented. Pythia-1B and Pythia-6.9B are established testbeds for memorization research (Biderman et al., 2023; Carlini et al., 2023). For Pythia, we use 50-token prefixes from 10,000 fixed sequences in each of four Pile subsets: Wikipedia, Enron, Pile-CC, and GitHub. We report text extraction for the first three subsets and code extraction for GitHub. For OLMo-2-13B, we use 50,000 Wikipedia sequences and 50,000 StarCoder sequences from OLMo-Mix-1124.

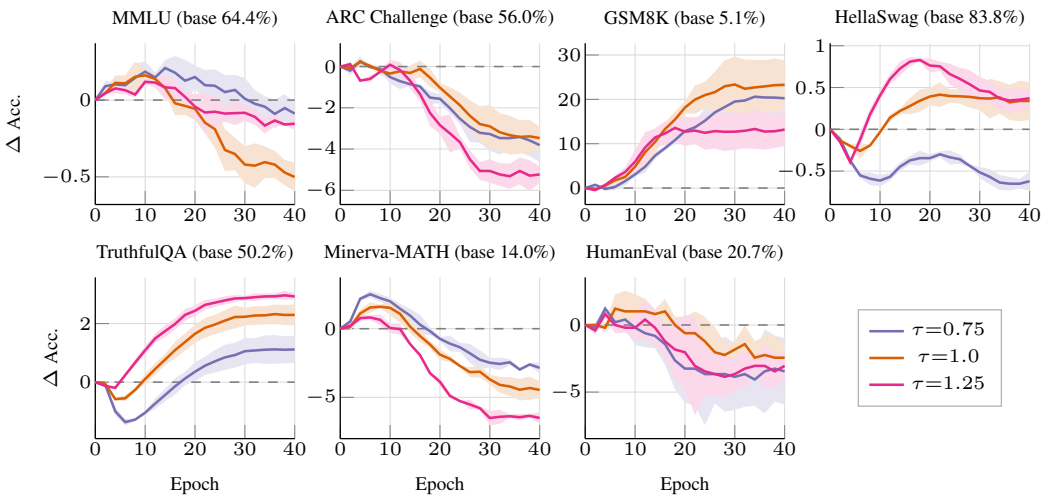


Figure 4: **On OLMo-2-13B, GSM8K and TruthfulQA improve over training. Minerva-MATH and HumanEval show early gains before falling below base, while ARC-Challenge declines.** Δ accuracy, in percentage points, relative to the base model over 40 epochs; shaded bands show ±1 std. across three training sets. GSM8K uses flexible-extract scoring and Minerva-MATH uses `math_verify`.

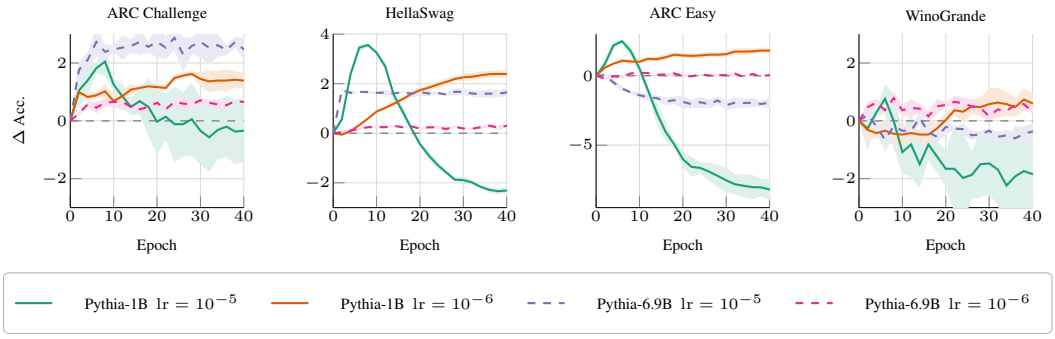


Figure 5: **Pythia gains depend on model size, learning rate, benchmark, and checkpoint; at learning rate 10^{-5}, later checkpoints lose accuracy on ARC-Easy and WinoGrande.** Δ accuracy, in percentage points, relative to the base model over 40 epochs on ARC-Challenge (25-shot), HellaSwag (5-shot), ARC-Easy (0-shot), and WinoGrande (5-shot). Shaded bands show ±1 std. across three training sets.

Extraction falls in six of the seven configurations (Figure 6). At a learning rate of 10^{-5}, text extraction falls by 97% on Pythia-1B and 94% on Pythia-6.9B, while code extraction roughly halves on both. At 10^{-6}, text extraction falls by 61% on Pythia-1B and remains essentially unchanged on Pythia-6.9B. On OLMo-2-13B, Wikipedia extraction falls by 29%, 52%, and 73% at τ=0.75, 1.0, and 1.25, respectively, and code extraction falls by 14%, 17%, and 23%. Higher temperatures therefore produce larger extraction reductions in these experiments. Text extraction declines more than code extraction in both model families, although the OLMo audit sets also differ in how they were constructed. The decline also appears under the alternative criteria for OLMo-2-13B Wikipedia sequences at τ=1.25, but the reductions are smaller. Near-verbatim extraction falls by 48%, and the number of sequences whose continuation probability is at least 10^{-3} falls by 40% (Appendix A).

The effect depends on model size and learning rate. At 10^{-5}, Pythia-6.9B reduces extraction sharply while keeping its gains on ARC-Challenge and HellaSwag, whereas Pythia-1B reduces extraction as much but loses utility later in training. Lower extraction can therefore accompany either improved or degraded benchmark performance. Conversely, at every evaluated epoch with a positive mean accuracy change, text extraction is at most 1% above the base extraction count. These aggregate results do not establish whether rehearsal contributes to the gains. On OLMo-2-13B, τ=1.25 produces

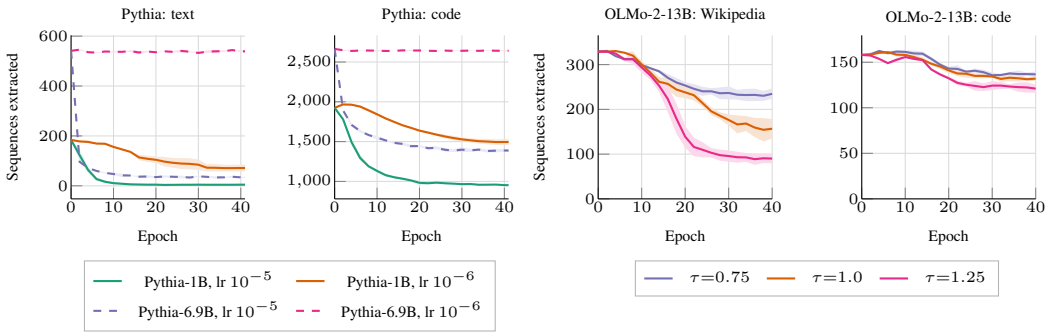


Figure 6: **Self-training lowers verbatim extraction of pretraining data in Pythia and OLMo-2-13B.** We count sequences whose true 50-token continuation is reproduced exactly from a 50-token prefix. Pythia: 30,000 text sequences (Enron, Pile-CC, Wikipedia) and 10,000 code sequences (GitHub) of the Pile, with shaded bands showing ± 1 std. across three training sets. OLMo-2-13B: 50,000 Wikipedia and 50,000 StarCoder sequences from OLMo-Mix-1124, with shaded bands across three training sets per temperature.

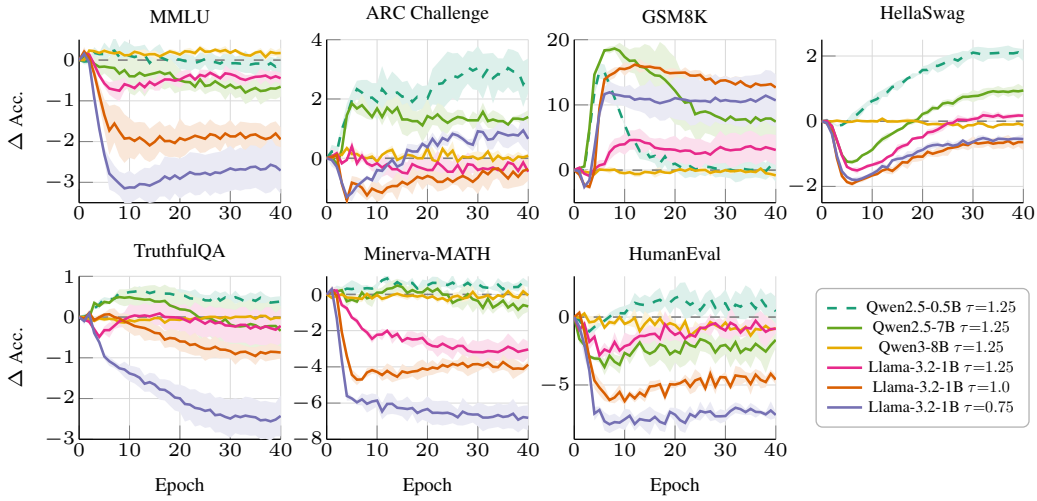


Figure 7: **The student's own samples give the largest gain on five of seven benchmarks, and samples from a different family often harm performance.** Δ accuracy, in percentage points, of the Qwen2.5-0.5B student over 40 epochs of training on corpora sampled from itself (dashed), Qwen2.5-7B, and Qwen3-8B (the Qwen3-8B-Base checkpoint) at τ=1.25, and from Llama-3.2-1B at three temperatures. Shaded bands show ± 1 std. across three training sets.

the largest extraction reduction, whereas τ=1.0 produces the largest GSM8K and mean benchmark gains. Extraction reduction is therefore not a proxy for utility.

6 Transfer depends on the source

Section 4 shows that prompt-free self-training can help, but only for some models and some tasks. We next ask whether source size and mean student NLL explain transfer to a fixed student. We keep the Qwen2.5-0.5B student, its training procedure, and the nominal corpus budget fixed. The sources are the student itself; Qwen2.5-7B (Yang et al., 2024), a larger model of the same lineage; Qwen3-8B-Base (Yang et al., 2025), the pretrained checkpoint of a larger model trained with a different recipe; and Llama-3.2-1B (Llama Team, AI @ Meta, 2024), another small model from a different family. We use τ=1.25 for the Qwen sources and vary temperature for Llama to examine whether sampling diversity changes transfer.

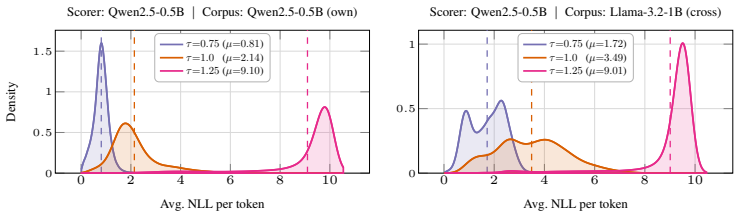


Figure 8: **Similar mean student NLL does not imply similar downstream transfer.** Kernel density of per-sample mean NLL per token for samples generated by Qwen2.5-0.5B (left) and Llama-3.2-1B (right), all scored by Qwen2.5-0.5B, with means over samples dashed. At τ=1.25 the two corpora have similar mean NLL, yet the student's own samples help far more (Figure 7).

We rank sources by the peak of each benchmark's mean curve (Figure 7). The student's own samples yield the largest gains on five of seven benchmarks. Qwen2.5-7B outperforms the tested Qwen3 and Llama sources on five of seven, even though Qwen3-8B is larger. Llama-generated data often lowers scores on MMLU, TruthfulQA, Minerva-MATH, and HumanEval, and varying its sampling temperature does not resolve these losses. Its effects are nevertheless task-dependent: at τ=1.0, Llama samples match the zero-shot GSM8K gain from self-generated data while lowering HumanEval. Source size alone therefore does not determine transfer under this procedure.

We next ask whether mean student NLL explains these transfer differences. For each sample in the self-generated and Llama corpora, we compute mean per-token NLL under Qwen2.5-0.5B, then average these values across samples (Figure 8). At lower temperatures, the student's own samples have lower NLL than Llama samples. At τ=1.25, their mean NLLs are similar: 9.10 nats per token for self-generated samples and 9.01 for Llama samples. Their benchmark trajectories nevertheless differ. Self-generated data improves ARC-Challenge, HellaSwag, and HumanEval, while Llama data produces smaller gains or losses on these tasks despite improving zero-shot GSM8K. The comparison also bears on accounts of RLVR that emphasize high-probability pretraining priors. Shao et al. (2025) argue that RLVR can bias updates toward these priors. In our setting, however, mean student NLL alone does not explain transfer: among the student's own corpora, the higher-NLL corpus at τ=1.25 can outperform the lower-NLL corpus at τ=0.75. Similar mean student NLL need not imply updates of similar magnitude or direction. Differences in these updates could explain the downstream effects, but we do not measure them here.

The source comparison motivates student-source compatibility as one possible explanation for transfer. Different corpora may induce updates that align differently with useful structure already learned by the student. Sources from the same lineage may generate sequences better matched to that structure, although we do not measure gradient alignment. Because the comparison uses one student, we also cannot establish whether the source ordering holds for other students. Under the resurfacing hypothesis, self-training helps when the synthetic corpus reinforces capabilities the student already supports.

7 Discussion and Limitations

At matched training epochs, six of the seven configurations achieve higher mean benchmark scores and lower mean extraction counts than their base models (Figure 1B and C). Pythia-6.9B at 10^{-6}, which changes little, is the exception. Individual benchmarks can still decline at these epochs. For OLMo-2-13B, the joint improvement also occurs in each replicate individually. For the three Pythia configurations with substantial extraction reductions, extracted text counts are already 61 to 85% below base at the epoch with the highest mean accuracy change. On OLMo-2-13B, the checkpoints with the lowest extraction still exceed the base model's mean benchmark score, mainly because of GSM8K and TruthfulQA. ARC-Challenge, Minerva-MATH, and HumanEval nevertheless finish below base.

Our training uses no forget set, privacy objective, or reward. The extraction reductions arise during untargeted training on the model's own samples. We do not directly compare with the targeted unlearning methods in Section 2, and lower extraction under these attacks does not establish that the underlying information has been removed.

Limitations. Our utility summaries report peaks selected on the evaluation benchmarks, and different benchmarks peak at different epochs. On OLMo-2-13B, Minerva-MATH and HumanEval improve early and then decline, while GSM8K and TruthfulQA remain near their peaks through epoch 40. Extraction continues to fall over training. No single checkpoint achieves every benchmark's maximum and the lowest extraction count, and selecting a useful stopping point without tuning on the evaluation benchmarks remains unresolved. We leave to future work a generation or training strategy that balances benchmark gains against extraction at a single checkpoint.

These results identify a regime in which mean benchmark scores improve while measured verbatim extraction falls. They are consistent with latent capability resurfacing, while leaving its mechanism unresolved. More broadly, our findings suggest that synthetic post-training should be designed around student-source compatibility, not synthetic data quality alone.

AI USE STATEMENT

We used generative AI tools to assist in implementing the codebase from author-specified designs and to edit portions of the manuscript from author-supplied content for language refinement and paraphrasing. We did not use generative AI tools to formulate the claims, design the methodology or experiments, or interpret the results. The authors reviewed all AI-assisted code and text and take full responsibility for the content of this work.

ETHICS STATEMENT

This work measures verbatim extraction of documented, publicly released pretraining data (the Pile and OLMo-Mix-1124) from open-weight models; we use existing publicly released datasets and collect no new private data. Lower extraction under our attacks is not a privacy guarantee: near-verbatim and probabilistic extraction fall less than exact match (Section 5), and we do not audit extraction under other attacks.

REPRODUCIBILITY STATEMENT

Section 3 describes the models, samples, training, and evaluation. Appendix D lists the key training and evaluation settings for the three training pipelines, and Appendix C describes how the extraction sets are built.

REFERENCES

Sina Alemohammad, Josue Casco-Rodriguez, Lorenzo Luzi, Ahmed Imtiaz Humayun, Hossein Babaei, Daniel LeJeune, Ali Siahkoohi, and Richard G. Baraniuk. Self-consuming generative models go MAD. In *International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=ShjMHfmPs0.

Sina Alemohammad, Ahmed Imtiaz Humayun, Shruti Agarwal, John Collomosse, and Richard G. Baraniuk. Self-improving diffusion models with synthetic data, 2024b. URL https://arxiv.org/abs/2408.16333.

Sina Alemohammad, Zhangyang Wang, and Richard G. Baraniuk. Neon: Negative extrapolation from self-training improves image generation. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=kpLRYtPGt3.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, Usvsn Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2397–2430. PMLR, 2023. URL https://proceedings.mlr.press/v202/biderman23a.html.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650. USENIX Association, 2021. URL https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramèr, and Chiyuan Zhang. Quantifying memorization across neural language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=TatRHT_1cK.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

A. Feder Cooper, Mark A. Lemley, Christopher De Sa, Lea Duesterwald, Allison Casasola, Jamie Hayes, Katherine Lee, Daniel E. Ho, and Percy Liang. Estimating near-verbatim extraction risk in language models with decoding-constrained beam search, 2026. URL https://arxiv.org/abs/2603.24917.

Yunzhen Feng, Elvis Dohmatob, Pu Yang, Francois Charton, and Julia Kempe. Beyond model collapse: Scaling up with synthesized data requires verification. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=MQXrTMonT1.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The Pile: An 800GB dataset of diverse text for language modeling, 2020. URL https://arxiv.org/abs/2101.00027.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, July 2024. URL https://zenodo.org/records/12608602.

Jamie Hayes, Marika Swanberg, Harsh Chaudhari, Itay Yona, Ilia Shumailov, Milad Nasr, Christopher A. Choquette-Choo, Katherine Lee, and A. Feder Cooper. Measuring memorization in language models via probabilistic extraction. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 9266–9291, 2025. URL https://aclanthology.org/2025.naacl-long.469/.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Matthew Jagielski, Om Thakkar, Florian Tramèr, Daphne Ippolito, Katherine Lee, Nicholas Carlini, Eric Wallace, Shuang Song, Abhradeep Guha Thakurta, Nicolas Papernot, and Chiyuan Zhang. Measuring forgetting of memorized training examples. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=7bJizxLKrR.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14389–14408, Toronto, Canada, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.805. URL https://aclanthology.org/2023.acl-long.805/.

Xiaotong Ji, Rasul Tutunov, Matthieu Zimmer, and Haitham Bou Ammar. Scalable power sampling: Unlocking efficient, training-free reasoning for LLMs via distribution sharpening. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2601.21590.

Aayush Karan and Yilun Du. Reasoning with sampling: Your base model is smarter than you think. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Vsgq2ldr4K.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626. Association for Computing Machinery, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Pierre-Carl Langlais, Pavel Chizhov, Catherine Arnett, Carlos Rosas Hinostroza, Mattia Nee, Eliot Krzystof Jones, Irène Girard, David Mach, Anastasia Stasenko, and Ivan P. Yamshchikov. Common Corpus: The largest collection of ethical data for LLM pre-training. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=0wSlFpMsGb.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems 35*, 2022. URL https://papers.nips.cc/paper_files/paper/2022/hash/18abbeef8cfe9203fdf9053c9c4fe191-Abstract-Conference.html.

Mengqi Li, Lei Zhao, Anthony Man-Cho So, Ruoyu Sun, and Xiao Li. A model can help itself: Reward-free self-training for LLM reasoning, 2025. URL https://arxiv.org/abs/2510.18814.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.acl-long.229. URL https://aclanthology.org/2022.acl-long.229/.

Llama Team, AI @ Meta. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C. Lipton, and J. Zico Kolter. TOFU: A task of fictitious unlearning for LLMs. In *First Conference on Language Modeling*, 2024. doi: 10.48550/arXiv.2401.06121. URL https://openreview.net/forum?id=B41hNBoWLo.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. ZeRO: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16, 2020. doi: 10.1109/SC41405.2020.00024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020.

Yan Scholten, Sophie Xhonneux, Leo Schwinn, and Stephan Günnemann. Model collapse is not a bug but a feature in machine unlearning for LLMs. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=1MCQzboBrR.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. Spurious rewards: Rethinking training signals in RLVR, 2025. URL https://arxiv.org/abs/2506.10947.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. AI models collapse when trained on recursively generated data. *Nature*, 631:755–759, 2024. doi: 10.1038/s41586-024-07566-y. URL https://www.nature.com/articles/s41586-024-07566-y.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, Abhishek Kumar, Alexander A. Alemi, Alex Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Fathy Elsayed, Hanie Sedghi, Igor Mordatch, Isabelle Simpson, Izzeddin Gur, Jasper Snoek, Jeffrey Pennington, Jiri Hron, Kathleen Kenealy, Kevin Swersky, Kshiteej Mahajan, Laura Culp, Lechao Xiao, Maxwell L. Bileschi, Noah Constant, Roman Novak, Rosanne Liu, Tris Warkentin, Yamini Bansal, Ethan Dyer, Behnam Neyshabur, Jascha Sohl-Dickstein, and Noah Fiedel. Beyond human data: Scaling self-training for problem-solving with language models. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=lNAyUngGFK.

Yuqiao Tan, Minzheng Wang, Bo Liu, Zichen Liu, Tian Liang, Shizhu He, Jun Zhao, and Kang Liu. From $P(y|x)$ to $P(y)$: Investigating reinforcement learning in pre-train space, 2026. URL https://arxiv.org/abs/2604.14142.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, et al. 2 OLMo 2 furious. In *Conference on Language Modeling*, 2025. URL https://arxiv.org/abs/2501.00656.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2024. URL https://arxiv.org/abs/2412.15115.

An Yang et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems 35*, pp. 15476–15488, 2022. URL https://papers.nips.cc/paper_files/paper/2022/hash/639a9a172c044fbb64175b5fad42e9a5-Abstract-Conference.html.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Ruixiang Zhang, Richard He Bai, Huangjie Zheng, Navdeep Jaitly, Ronan Collobert, and Yizhe Zhang. Embarrassingly simple self-distillation improves code generation, 2026. URL https://arxiv.org/abs/2604.01193.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 400–410, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-demos.38. URL https://aclanthology.org/2024.acl-demos.38/.

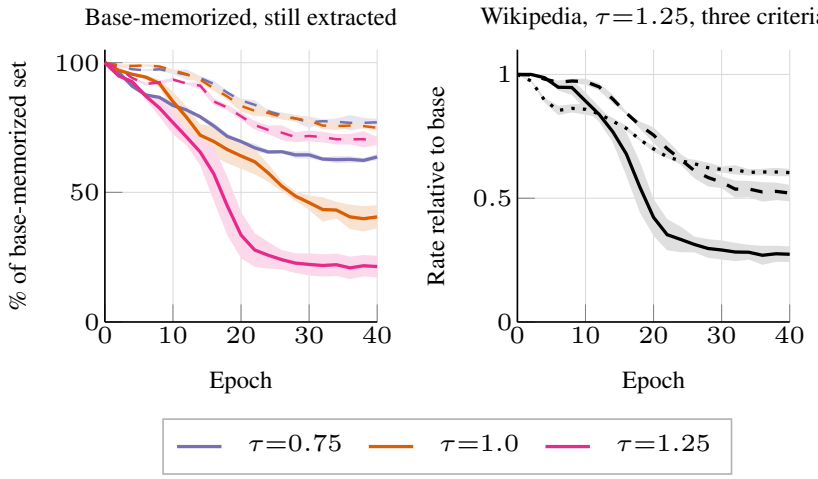


Figure 9: **On OLMo-2-13B, many sequences extracted from the base model are no longer extracted after training. Counts under the alternative extraction criteria decline less than exact-match counts.** Left: share of the sequences the base model reproduces exactly (329 Wikipedia, solid; 158 code, dashed) that each checkpoint still reproduces. Right: at τ=1.25, near-verbatim extraction (dashed, edit distance ≤ 5) and probabilistic extraction (dotted, probability $\geq 10^{-3}$) relative to base, against exact match (solid). Shaded bands show ± 1 std. across three training sets.

A Retention and extraction criteria on OLMo-2-13B

Figure 9 tracks the sequences extracted exactly from the OLMo-2-13B base model and compares the three extraction criteria at τ=1.25. Section 5 reports the numerical reductions. Figure 6 counts extracted sequences across the full audit set; the left panel of Figure 9 follows only sequences extracted from the base model and therefore excludes newly extractable sequences.

B Few-shot evaluation of OLMo-2-13B

Figure 10 evaluates the OLMo-2-13B checkpoints of one training set per temperature under the few-shot protocols used for Qwen2.5-0.5B: ARC-Challenge 25-shot, HellaSwag 5-shot, GSM8K 8-shot with flexible-extract scoring, and Minerva-MATH 4-shot with `math_verify`. ARC-Challenge and HellaSwag gain by up to 1.5 and 1.8 points and end above base at epoch 40 at every temperature. GSM8K and Minerva-MATH gain by at most 1.2 and 0.7 points in the first few epochs and then fall below base. At epoch 40, GSM8K is 11.6, 12.0, and 20.6 points below its 8-shot base accuracy of 72.8% at τ=0.75, 1.0, and 1.25, and strict-match scoring shows the same decline. The zero-shot GSM8K gain of Section 4 therefore does not carry over to 8-shot evaluation on OLMo-2-13B. With a single training set per temperature, the figure shows no variation across training sets.

C Extraction sets

Pythia. 10,000 fixed sequences from each of the Wikipedia, Enron, Pile-CC, and GitHub subsets of the Pile, with a 50-token prefix and the true continuation as target.

OLMo-2-13B. Each OLMo sequence contains 100 tokens: a 50-token prefix and a 50-token continuation. For Wikipedia, we take the first 100 tokens of Dolma Wikipedia documents in OLMo-Mix-1124. For code, we take one random 100-token window from each StarCoder file in OLMo-Mix-1124, with boilerplate filtered. Each set contains 50,000 sequences in random order. We supply prefixes as token IDs without a beginning-of-sequence token and force generation to exactly 50 tokens. The base-memorized set contains every sequence reproduced exactly by the base model. For probabilistic extraction, we compute the teacher-forced continuation probability under top-40 renormalization.

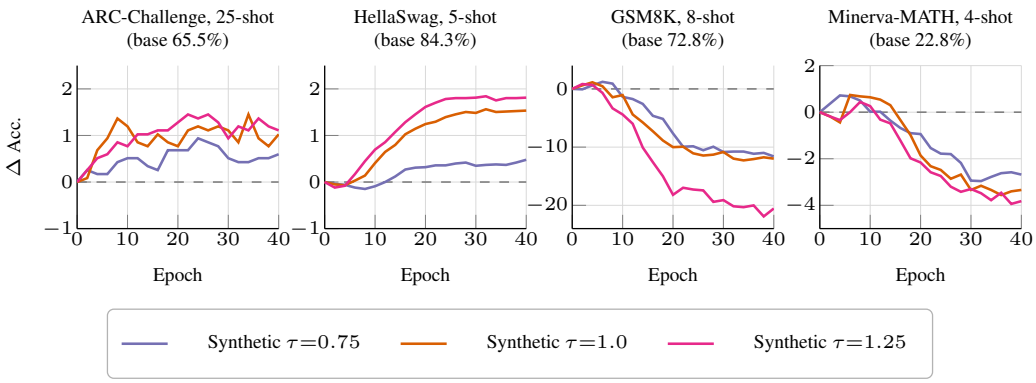


Figure 10: **Under few-shot evaluation, OLMo-2-13B gains on ARC-Challenge and HellaSwag, while GSM8K and Minerva-MATH fall below base after early training.** Δ accuracy, in percentage points, relative to the base model over 40 epochs under ARC-Challenge 25-shot, HellaSwag 5-shot, GSM8K 8-shot (flexible-extract), and Minerva-MATH 4-shot (`math_verify`) evaluation, for one training set per temperature.

D Reproducibility Details

All experiments use NVIDIA A100 80GB GPUs.

Qwen2.5-0.5B. Generation uses vLLM (Kwon et al., 2023) in bf16 from a single newline token (Qwen2.5 has no beginning-of-sequence token), with `top_p` $= 1.0$, `top_k` $= -1$, `min_tokens` $=$ 50, `max_tokens` $= 1024$, and master seed 42 (seed $42 + 1000g$ on GPU g). We clear the model-default stop tokens because vLLM's `min_tokens` does not block them. The 8-gram filter removes a sample if it shares at least one 8-gram with the train, validation, or test split of ARC, GSM8K, HellaSwag, MMLU, TruthfulQA, MATH, or HumanEval. We construct each of the three training sets by sampling uniformly with replacement from the filtered pool until it reaches the nominal 5M-token budget, estimated at four characters per token. The sets can therefore share samples. We use LLaMA-Factory (Zheng et al., 2024) for continued pretraining with DeepSpeed ZeRO-3 (Rajbhandari et al., 2020) in bf16. We fine-tune all parameters with AdamW and framework-default optimizer hyperparameters. Training uses a learning rate of 10^{-6}, an effective batch size of 64, and a cosine schedule with 10% warmup that decays to zero. We pack sequences to 2,048 tokens with an empty template, train for 40 epochs, and save a checkpoint after each epoch. Evaluation uses lm-evaluation-harness (Gao et al., 2024) with the vLLM backend in bf16 and seed 42, with greedy decoding for every generation task except HumanEval, where pass@1 is estimated from 10 samples per problem at temperature 0.2. Each replicate takes about 4 GPU-hours.

Pythia. Training uses HuggingFace Transformers in fp32 with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0, gradient clipping 1.0), a linear warmup-then-decay schedule with 50 warmup steps, effective batch size 32 (4×8 accumulation), sequences of 1,024 tokens, learning rates $\{10^{-5}, 10^{-6}\}$, and 40 epochs. Generation uses HuggingFace Transformers in fp16 at τ=1.25 with the same length limits, and the same 8-gram filter removes samples that overlap ARC-Challenge, ARC-Easy, HellaSwag, or WinoGrande. Each configuration has three replicates, one per training set. Models are the `main` revisions of `EleutherAI/pythia-1b` and `EleutherAI/pythia-6.9b`. Extraction evaluation uses fp16 with greedy decoding. Each Pythia-1B replicate takes about 8 GPU-hours and each Pythia-6.9B replicate about 24.

OLMo-2-13B. Training uses the HuggingFace Trainer with DeepSpeed ZeRO-3 in bf16, full-parameter AdamW ($\epsilon = 10^{-8}$, weight decay 0, gradient clipping 1.0), a cosine schedule with 10% warmup, learning rate 10^{-6}, effective batch size 64, sequence packing at 2,048 tokens with an end-of-text token after each sample, gradient checkpointing, and 40 epochs with a checkpoint every 2 epochs. Generation uses vLLM from `<|endoftext|>`, with the same length limits and no truncation, until the pool holds 10M tokens; the same 8-gram filter as for Qwen2.5-0.5B removes samples that overlap any of the seven benchmarks, and three 5M-token subsets per temperature, drawn with different random seeds, are used for training. Evaluation uses lm-evaluation-harness with

the vLLM backend, zero-shot, with the same decoding as for Qwen2.5-0.5B. Extraction uses vLLM with token-id prompts. The model is `allenai/OLMo-2-1124-13B`.