OFF-POLICY MERGING BEATS ON-POLICY SELF-DISTILLATION FOR CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

A long-standing goal of AI is a model that can continually learn and improve itself. On heavily post-trained models, supervised finetuning (SFT) on new data often causes rapid forgetting. As such, it is commonly believed that *on-policy* training is a prerequisite to prevent forgetting. In practice, however, data containing novel information or experiences are usually *off-policy*. While methods such as On-Policy Self-Distillation (OPSD) have been proposed to bridge this gap by converting off-policy data into on-policy signal, they have been demonstrated to cause reasoning collapse. In this paper, we show that *embarrassingly simple off-policy training beats OPSD for continual learning*. We first premise that the issue with SFT is interference with existing capabilities, and worse, the same SFT produces progressively more interference on checkpoints later along the model's training trajectory. However, we demonstrate that the drawbacks of SFT can be almost entirely circumvented with a simple recipe we term **grafting**, consisting of the following: (1) **donor checkpoints:** training the weight update from an earlier checkpoint and applying the weight *delta*, (2) **scaling** the weight delta, equivalent to a form of weight-averaging, and (3) **masking** sensitive update directions when the data is too off-policy. Across injecting new knowledge, learning from expert traces, and self-improvement, our off-policy grafting method Pareto-dominates both SFT and OPSD over injected and retained capabilities, while avoiding expensive on-policy sampling. Consequently, our work challenges on-policy training as a necessity for continual learning on RL-trained models.

1 INTRODUCTION

A long-standing goal of AI is a model that keeps acquiring new capabilities without forgetting what it can already do (McCloskey & Cohen, 1989; Kirkpatrick et al., 2017). This is particularly challenging for models that have undergone extensive post-training. In particular, direct supervised finetuning (SFT) on new data can cause forgetting and fail to generalize (Chu et al., 2025). This has motivated the view that continual learning should remain *on-policy*, i.e. the model learns from behavior produced by its current policy (Chen et al., 2025). However, extended on-policy RL can plateau and lose coverage after exhausting non-zero reward behaviors to amplify (Yue et al., 2025; Zhang et al., 2026). Furthermore, it is often desirable for models to learn from *off-policy* data, such as new knowledge after the pretraining cutoff (Jang et al., 2022), expert demonstrations (Taori et al., 2023), delayed feedback from environments (Trajectory Labs, 2026), and synthetic data for self-improvement (Chakraborty et al., 2026).

Recent work on on-policy self-distillation (OPSD) tries to bridge this gap by converting off-policy data into on-policy updates, distilling from a solution-conditioned copy of the model (Zhao et al., 2026; Hubotter et al., 2026; Shenfeld et al., 2026). However, conditioning on the solution can suppress exploration, leak privileged information into the teacher's reasoning, and degrade out-of-domain performance or collapse reasoning length (Kim et al., 2026; Kaur et al., 2026). More fundamentally, this raises a simple question:

Is on-policy necessary for continual learning, or are we using off-policy updates poorly?

We study three settings where off-policy data arises naturally: (1) **learning from expert traces** on hard problems the model cannot solve; (2) **self-improvement**, where models bootstrap themselves without stronger supervision, e.g., via their own rejection-sampled successes (Zelikman et al.,

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.

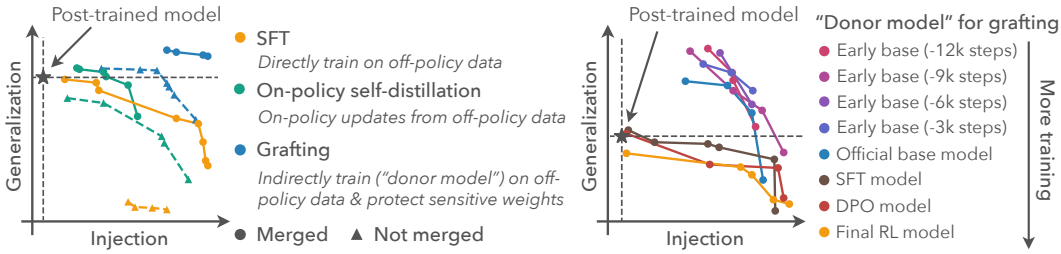


Figure 1: **(Left) Model merging alone greatly improve generalization; Grafting dominates SFT and OPSD.** Model: Qwen3-1.7B (thinking). **(Right) Grafting from earlier checkpoints works better, from pretraining to post-training.** Model: OLMo-3-7B (thinking).

2022) or from traces written by a copy of themselves with privileged information (Pedagogical RL; Chakraborty et al., 2026), and (3) **knowledge injection** of new facts absent from prior training. In all three, we assume access only to the data, not to the weights or logits of a stronger teacher model, so on-policy distillation (OPD) from a teacher (Agarwal et al., 2023) is not possible.

We premise with a surprisingly simple finding: model merging, i.e., by interpolating the SFT-ed weights back toward the original post-trained model (Wortsman et al., 2022b), $\theta_{\text{merge}} = (1 - \lambda)\,\theta_{\text{post}} + \lambda\,\theta'_{\text{post}}$ where θ'_{post} is the model after SFT on a task, SFT is elevated to a strong baseline (Figure 1, left), achieving higher performance on the new task and comparable performance on old tasks. Since merging only scales the SFT update down without changing its direction, this demonstrates SFT learns a useful signal from the new data. However, naive integration of SFT updates clearly interferes with the post-trained model's capabilities. This leads to a tangible key objective:

How can we better minimize the interference induced by off-policy updates?

We first rewrite merging as adding a scaled update to the post-trained model, $\theta_{\text{merge}} = \theta_{\text{post}} + \lambda\,(\theta'_{\text{post}} - \theta_{\text{post}})$. Written this way, merging decouples two design choices: *where* the update is learned (here, naively on θ_{post} itself) and *how* it is applied (here, naively scaled by λ).

For the *where*: task arithmetic (Ilharco et al., 2023) shows that an update need not be derived from the model it is applied to. Why is this separation important for continual learning? Training on a narrow distribution skews which task a model infers its inputs to come from (Kotha et al., 2024), so finetuning the post-trained model routes much of the learning signal through the parameters behind its existing capabilities, while a checkpoint from before post-training routes less of it there (§G). We therefore learn the update on an earlier donor model θ_{donor}, and apply it to the post-trained model: $\theta_{\text{post}} + \lambda\,(\theta_{\text{donor-ft}} - \theta_{\text{donor}})$, where $\theta_{\text{donor-ft}}$ is the donor after finetuning. Along the OLMo-3 training trajectory (OLMo Team, 2026), we see that the best donors are checkpoints before the end of pretraining (Figure 1, right).

For the *how*: donor-trained updates write less to the parameters on which the post-trained model is most sensitive (Figure 7, right), but when the new distribution is far from the post-trained model, they can still damage them. We therefore also protect the most sensitive directions, estimated with the Fisher information (Kirkpatrick et al., 2017; Ikram et al., 2026). This matters most for knowledge injection (§5.2), where retaining unrelated capabilities is hard. We call the full recipe **grafting**: $\theta_{\text{post}} + \lambda \cdot \mathcal{M}_\rho(\theta_{\text{donor-ft}} - \theta_{\text{donor}})$, where the mask $\mathcal{M}_\rho$ projects away the top-ρ fraction of Fisher-weighted directions from the update.

In all three problem settings, grafting dominates OPSD in both new-task and old-task performance while avoiding expensive on-policy sampling. In self-improvement, grafting roughly doubles new-task performance over the post-trained model while also improving old-task performance. Together, our results suggest that on-policy updates are not a necessary condition for continual learning. Since continual learning applications often involve off-policy experiences, we hope our work inspires more effort on making off-policy training generalize.

2 PRELIMINARIES

We consider teaching a post-trained model $\pi_{\theta_{\text{post}}}$ a new target task, formally a set of inputs $\{x\}$ which we want the model to solve. A task may be a skill (e.g., a class of hard math problems) or some knowledge (e.g., new facts after the knowledge cutoff). Two objectives should be optimized jointly:

- **New task**: performance on held-out IID instances of the target task.
- **Old task**: performance on tasks outside the target task – for unrelated tasks, we want the capability to be retained (i.e., less forgetting); for related tasks, we want the new skill or knowledge to generalize to these tasks.

Reinforcement learning (RL) is a standard way to learn a task without using off-policy data. It maximizes $\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\, R(x,y)$ over outputs it samples itself. However, given its mode-seeking nature, RL is known to mostly amplify and compose behaviors the model already possesses (Zhang et al., 2026). For genuinely new knowledge or very hard problems beyond the model's capability, the model places almost no probability on correct outputs and, therefore, receives no training signal to continue improving. These are the scenarios where off-policy data can be extremely useful.

Off-policy data can be formalized as a corpus $\mathcal{D} = \{(x, y)\}$ of inputs x and outputs y, where x are not generated under $\pi_{\theta_{\text{post}}}$. Such data is abundant in practice, e.g., new knowledge acquired after the model's pretraining cutoff, expert demonstrations for a hard task, feedback from new environments, and synthetic data created by the model for self-improvement.

Supervised fine-tuning (SFT) has been the standard way to learn from off-policy data, optimizing the model likelihood $-\mathbb{E}_{(x,y)\sim\mathcal{D}} \log \pi_\theta(y \mid x)$ directly on the outputs y given by the data $\mathcal{D}$. However, SFT is known to memorize rather than generalize (Chu et al., 2025) and to cause catastrophic forgetting of capabilities outside $\mathcal{D}$ (McCloskey & Cohen, 1989; Kirkpatrick et al., 2017; Biderman et al., 2024).

On-policy self-distillation (OPSD) addresses the shortcomings of SFT by converting the off-policy corpus into on-policy updates (Zhao et al., 2026; Shenfeld et al., 2026; Hubotter et al., 2026). The model samples responses and is supervised at each token by a solution-conditioned version of itself:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}\, \mathbb{E}_{\tilde{y}\sim\pi_\theta(\cdot|x,\,y)} \sum_t \mathrm{D}_\alpha\big(\pi_{\bar{\theta}}(\cdot \mid x, y, \tilde{y}_{<t}) \,\big\|\, \pi_\theta(\cdot \mid x, \tilde{y}_{<t})\big),$$

where the teacher $\bar{\theta}$ is a stop-grad copy of the student , and D_α is a family of f-divergences (Agarwal et al., 2023) whose α interpolates from forward KL (α=0) to reverse KL (α=1).

3 METHOD

We recall that **grafting** consists of two main components: a scaled parameter delta derived from training on a *donor* model (i.e., a checkpoint from earlier in training), and a *projection* mask.

Extracting updates from a donor model. Instead of directly finetuning the post-trained model θ_{post}, we revisit task arithmetic (Ilharco et al., 2023) (or chat vector (Huang et al., 2024), Shadow-FT (Wu et al., 2025)), which shows that weight updates induced by finetuning the base model can be combined with later checkpoints. We generalize this to finetuning any earlier checkpoint which we call a "donor model" θ_{donor}. Specifically, we finetune on the donor model θ_{donor} to obtain $\theta_{\text{donor-ft}}$, and isolate the difference $\Delta = \theta_{\text{donor-ft}} - \theta_{\text{donor}}$. We may then apply the update scaled by λ to θ_{post}:

$$\theta(\lambda) \;=\; \theta_{\text{post}} \;+\; \lambda\,\big(\theta_{\text{donor-ft}} - \theta_{\text{donor}}\big). \tag{1}$$

While $\lambda \in (0, 1]$ controls the scale of the graft, we remark that λ admits an interpretation that goes beyond interpolating between old and new task performance, arising from the *weight-averaging* literature (see e.g., `WiSE-FT` (Wortsman et al., 2022b; Dang et al., 2025)), where linearly interpolating a later checkpoint with an earlier one $\theta_{\text{WA}} = \lambda\theta + (1-\lambda)\theta_{\text{base}}$ is known to benefit reasoning and generalization. In particular, $\theta(\lambda)$ can be viewed as weight-averaging $\theta_{\text{donor-ft}}$ and θ_{donor} and using the interpolant $\theta_{\text{WA}} = \lambda\theta_{\text{donor-ft}} + (1-\lambda)\theta_{\text{donor}}$ as the graft source:

$$\theta_{\text{post}} + (\theta_{\text{WA}} - \theta_{\text{donor}}) = \theta_{\text{post}} + \lambda(\theta_{\text{donor-ft}} - \theta_{\text{donor}}) = \theta(\lambda).$$

This suggests that moderate λ can be beneficial for reasoning-intensive domains, which is reflected by our experiments on grafting from hard math data. However, though reduced compared to standard SFT, donor-derived updates can introduce interference especially for larger values of λ.

Protect sensitive weights via projection As we will see, applying a donor-derived update directly may not be enough when the injected distribution is far from the post-trained model (e.g., injecting new facts into a reasoning model). However, precisely characterizing the parameter directions responsible for interference is complex. We take inspiration from a motivation of on-policy methods, and consider a surrogate goal of shaping the update direction such that the **updated model modifies the behavior on the old domain as little as possible**. Concretely, we want an update Δ that keeps the conditional KL divergence $\mathbb{E}_{z\sim\mathcal{D}_{\text{old}}}\left[D_{\text{KL}}(p_{\theta_{\text{post}}}(\cdot\,|\,z)\;\|\;p_{\theta_{\text{post}}+\Delta}(\cdot\,|\,z))\right]$ over the old domain small. To efficiently translate this to the directions of Δ to mask, we recall the standard quadratic approximation of KL (Cover et al., 1991):

$$D_{\text{KL}}(p_{\theta_{\text{post}}}(\cdot\,|\,z)\;\|\;p_{\theta_{\text{post}}+\Delta}(\cdot\,|\,z)) \approx \frac{1}{2}\Delta^\top F_z(\theta_{\text{post}})\Delta, \tag{2}$$

where $F_z(\theta_{\text{post}}) = \mathbb{E}_{y\sim p_{\theta_{\text{post}}}(\cdot\,|\,z)}\Big[\big(\nabla_\theta \log p_{\theta_{\text{post}}}(y\,|\,z)\big)^{\otimes 2}\Big]$ is the conditional Fisher Information evaluated at θ_{post}. This can be estimated with samples over a retention set $\mathcal{D}_{\text{ret}} = \{z_1, \ldots, z_N\}$ of inputs from, e.g., an old domain:

$$F_{\text{ret}} := \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{y\sim p_{\theta_{\text{post}}}(\cdot\,|\,z_n)}\left[\Big(\nabla_\theta \log p_{\theta_{\text{post}}}(y\,|\,z_n)\Big)^{\otimes 2}\right]. \tag{3}$$

However, the full-parameter Fisher is infeasible to compute. We consider existing approaches that take a diagonal (i.e. entrywise) approximation (Kirkpatrick et al., 2017). We also propose another efficient approximation informed by the *Kronecker-Factored Approximate Curvature* (KFAC) (Martens & Grosse, 2015), similar to approaches in model-editing literature (Ikram et al., 2026), which we can have achieve better performance with less masking than the diagonal approximation. For concreteness, we walk through deriving the mask from the diagonal approximation. Measuring the Fisher quadratic form along the diagonal: $\Delta^\top \operatorname{diag}(F_{\text{ret}})\Delta = \sum_{j=1}^{d_\theta}(F_{\text{ret}})_{jj}\Delta_j^2$, we observe the entry-wise score $s_j = (F_{\text{ret}})_{jj}\,\Delta_j^2$ **approximates how much entry j in the update Δ contributes to the model change** (2). Therefore, a natural approach to reducing the model change induced by Δ is to *mask* the top ρ-fraction of entries S_ρ ranked by s_j:

$$\theta(\lambda,\rho) \;=\; \theta_{\text{post}} \;+\; \lambda\,(\mathcal{M}_\rho \odot \Delta), \qquad (\mathcal{M}_\rho)_j \;=\; \begin{cases} 0 & j \in S_\rho, \\ 1 & \text{otherwise}, \end{cases} \tag{4}$$

where $\odot$ is the element-wise product. We defer analogous derivation of the KFAC-based projection and other implementation details in this section to Appendix F. In our experiments, we show that retention-based projections derived from the Diagonal or KFAC Fisher work remarkably well in removing interference from donor updates by just masking a small number of directions ($\rho \leq 0.01$), outperforming prior Fisher-based merging (Matena & Raffel, 2022), and pushing the old/new-domain performance frontier compared to un-projected updates.

4 Experiments on Learning Hard Math Problems

4.1 Experimental setup

Models We experiment with two model pairs: Qwen3-1.7B (thinking) / Qwen3-1.7B-Base (Yang et al., 2025) and Qwen3-8B (thinking) / Qwen3-8B-Base as the post-trained model θ_{post} and the donor model θ_{donor}. Our experiments include both thinking and instruct modes. Unless stated otherwise the donor model is the officially released base model of the same family. In §4.3 we additionally use OLMo-3-7B (thinking) (OLMo Team, 2026) as θ_{post}, because it releases intermediate checkpoints along both its pretraining and post-training trajectories, which lets us vary θ_{donor} rather than fix it to the official base model.

Math data We consider three settings for where the off-policy data comes from (see §A):

- *Teacher traces*: the AoPS subset of Nemotron-SFT-Math-v4 (NVIDIA, 2026; Du et al., 2025), whose solutions are generated by DeepSeek-V4-Pro (DeepSeek-AI, 2026). At both scales we mine hard problems from this pool, so that the injected skill is one the post-trained model does not already have. For each model, we sample 16 attempts per problem and keep those with 0 success for 1.7B and ≤ 1 success for 8B, leaving 1,533 training samples for 1.7B and 985 for 8B.

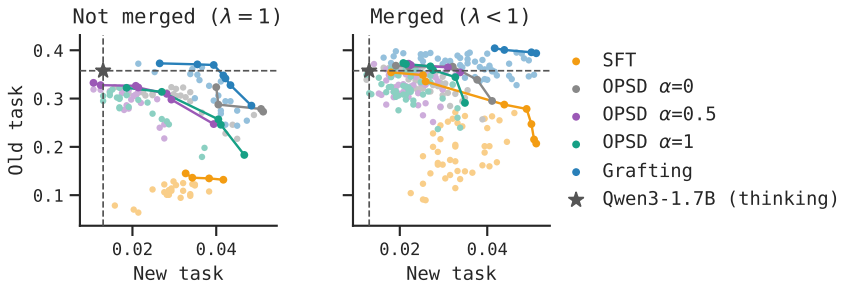


Figure 2: **Model merging improves generalization.**

- *Self-improvement with rejection sampling* (STaR; Zelikman et al., 2022): we apply rejection sampling to the post-trained model to get supervision of both thinking and solution tokens. We sample up to 32 attempts per problem and keep one verified trace for each solved problem.
- *Self-improvement with Pedagogical RL* (Chakraborty et al., 2026): STaR requires that the model can solve the problem with a large budget. Pedagogical RL removes this requirement by synthesizing reasoning traces from gold solutions and do SFT on the synthesized traces (Appendix B). We find replacing the SFT step of Pedagogical RL with grafting largely improves performance.

Evaluation New-task performance is measured on held-out data from the training set. For the math task, it is the accuracy on around 100 held-out problems. Old-task performance is the mean over three benchmarks (AIME 2024, AIME 2025, HMMT 2025). All metrics are estimated over 16 samples at temperature 1.0 and top-p 1.0, decoding up to 32K tokens in thinking mode, unless otherwise specified. Appendix C gives the training details.

4.2 Model merging alone improves generalization

We first show that model merging already improves generalization. We consider three methods: SFT, OPSD, and grafting, with full details given in Appendix C. In Figure 2, we sweep all methods with 9 learning rates (5×10^{-7} to 5×10^{-5}) and train 5 epochs. Figure 2 (left) shows that, without model merging, neither SFT nor OPSD outperforms the post-trained model, while grafting with the best hyperparameters does. We then sweep the merging scale for all methods in Figure 2 (right). We see that merging improves generalization and erases OPSD's original edge over SFT, despite its slow and expensive on-policy sampling. Given its uniform benefit, we apply merging across all methods in the rest of the paper.

4.3 Tracking θ_{donor} along pretraining and post-training with OLMo-3

OLMo-3-7B (thinking) (OLMo Team, 2026) has many checkpoints for us to move θ_{donor} freely. On the post-training side, we have base model $\xrightarrow{\text{SFT}}$ SFT checkpoint $\xrightarrow{\text{DPO}}$ DPO checkpoint $\xrightarrow{\text{RL}}$ OLMo-3-7B (thinking); on the pretraining side, we move θ_{donor} up to 12k steps before the base model.

We first investigate how each post-training stage makes finetuning harder. We see in Figure 3 (left) that grafting (base ckpt) outperforms grafting (SFT ckpt), grafting (SFT ckpt) outperforms grafting (DPO ckpt), and grafting (DPO ckpt) outperforms direct SFT on the final post-trained model. This suggests that each stage of post-training (SFT, DPO, RL) makes OLMo-3 harder to finetune.

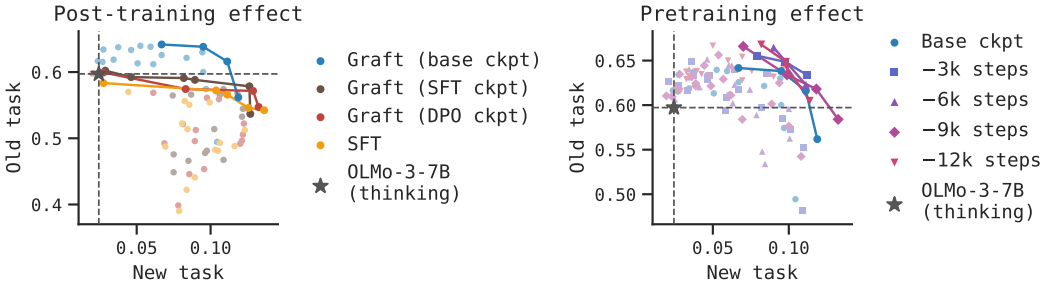
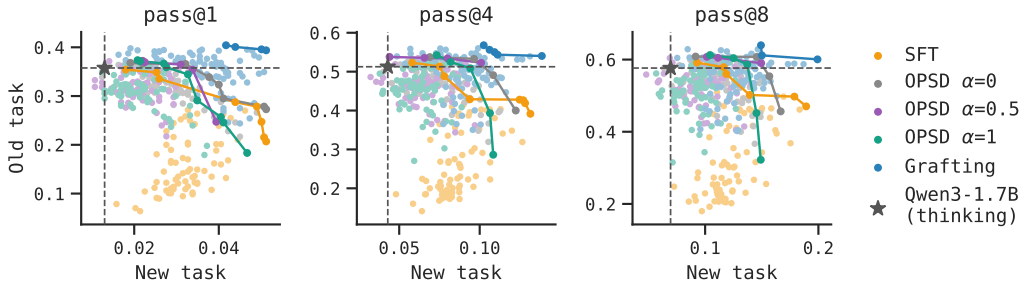
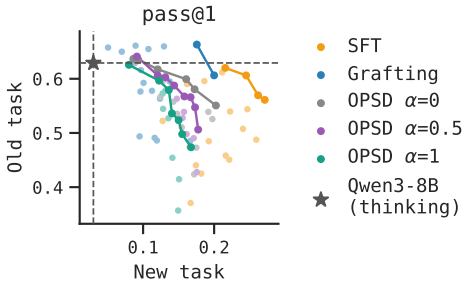


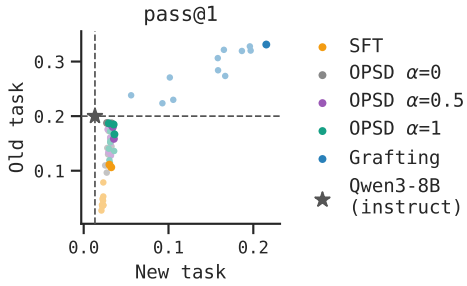
Figure 3: **Grafting from different θ_{donor} with OLMo-3-7B (thinking).** (Left) θ_{donor} is varied along the post-training trajectory: base model $\xrightarrow{\text{SFT}}$ SFT checkpoint $\xrightarrow{\text{DPO}}$ DPO checkpoint $\xrightarrow{\text{RL}}$ OLMo-3-7B (thinking). (Right) θ_{donor} is moved back along pretraining, up to 12k steps before the base model.



(a) Qwen3-1.7B (thinking)



(b) Qwen3-8B (thinking)



(c) Qwen3-8B (instruct)

Figure 4: **Results using hard-math teacher traces as off-policy data** across different models.

The question that remains is which pretraining checkpoint to use as the donor model. There is a tension in this choice. The end of pretraining anneals on higher-quality data and already resembles an early stage of SFT, so an earlier θ_{donor} should be easier to finetune, but it also sits farther from θ_{post}, which makes it less likely to stay linearly connected (Frankle et al., 2020; Neyshabur et al., 2020) with θ_{post}. Figure 3 (right) moves θ_{donor} up to 12k steps before the final base checkpoint. We see that earlier pretrained checkpoints clearly outperforms the final pretrained checkpoint, while the gain is not strictly monotonic. This suggests that for grafting we should search for an early pretrained checkpoint that balances the two forces. Qwen3 releases only its final base model and no earlier pretraining checkpoints, so in the rest of the paper we use the official base model as θ_{donor}.

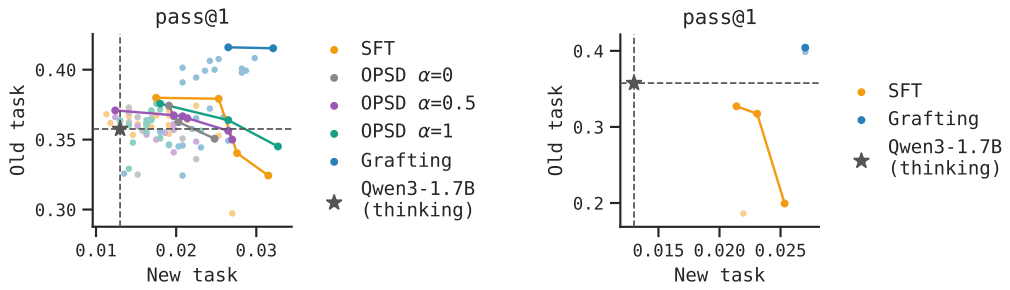
4.4 Math results with teacher traces

Thinking mode Figure 4a and Figure 4b show results with teacher traces as off-policy data, on Qwen3-1.7B (thinking) and Qwen3-8B (thinking), respectively. The same merging sweep (§4.2) is applied for all methods. We see that, for 1.7B, grafting dominates both SFT and OPSD and is the only method that meaningfully dominates the post-trained model. The same ordering holds at pass@4 and pass@8, suggesting that the gain reflects expanded capability and not simply a "sharpening effect". For 8B, both grafting and SFT dominate OPSD. For both sizes, OPSD fails to largely outperform the original post-trained model for generalization, which echoes the findings in Kim et al. (2026); Kaur et al. (2026).

Non-thinking mode (aka. instruct mode) We repeat the experiment at 8B in *non-thinking* mode on DAPO (Yu et al., 2025), an easier data for the non-thinking mode to learn, and we follow the thinking mode experiments to mine hard problems (see Appendix A for details). The same merging sweep (§4.2) is applied for all methods. We see from Figure 4c that grafting again dominates SFT and OPSD at pass@1.

4.5 Self-improvement for math

We now turn to the second data setting, where the off-policy data come from the post-trained model itself via rejection sampling (§4.1), also known as STaR (Zelikman et al., 2022). Unlike learning from teacher traces, STaR lets the model self-improve by training on its own data. Figure 5a shows that injection roughly doubles with grafting over the baseline while generalization rises about six points above it. The gain of SFT (i.e., the original STaR) is smaller. This indicates grafting has great potential in self-improvement, where models bootstrap themselves without stronger supervision.



(a) Rejection sampling (STaR) (b) Pedagogical RL

Figure 5: **Self-improvement results with Qwen3-1.7B (thinking).**

Pedagogical RL (Chakraborty et al., 2026) takes a different route to self-improvement: a copy of the model that sees the reference solution is trained with RL to write traces the model can learn from (Appendix B), and the original method then learns these traces with SFT. Figure 5b shows that on the same traces, grafting again dominates SFT: grafting keeps old-task performance above the post-trained model, while SFT loses old-task performance with merging and collapses without it.

5 Experiments on Knowledge Injection

5.1 Experimental setup

Models We experiment with Qwen3-1.7B (thinking) as the post-trained model θ_{post} and Qwen3-1.7B-Base (Yang et al., 2025) as the donor model θ_{donor}.

Knowledge data Our data for knowledge acquisition mainly follows Shenfeld et al. (2026), but with the key exception that we use purely synthetic knowledge to avoid any possible leakage. We construct a fictional world with 200 invented entities and 1,125 atomic facts. Each fact is written as question-answer pairs in 8 distinct phrasings, and the model is trained on those; evaluation then uses *held-out paraphrases* the model has never seen. For OPSD, we write wiki-style articles that state the fictional facts for conditioning. Since the knowledge data's QA form is very different from a reasoning model's, we mix knowledge QA with AoPS teacher traces we use in the math experiments. Together, we have 9,000 QA samples and 1,533 AoPS traces. All methods are trained on this same mixture.

Evaluation New-task performance is measured on held-out data from the training set. For knowledge, it is the exact-match accuracy on held-out paraphrases the model never saw, decoded greedily. Old-task performance is the mean over three benchmarks (AIME 2024, AIME 2025, HMMT 2025). Appendix C gives the training details.

5.2 Knowledge injection results

Figure 6 (left) shows the knowledge injection result with Qwen3-1.7B (thinking), where the same model merging sweep (§4.2) is applied to all methods. We see that grafting achieves nearly perfect injection with retention at or above the post-trained model. SFT injects just as well but suffers from forgetting. OPSD achieves near-perfect retention but also learns less.

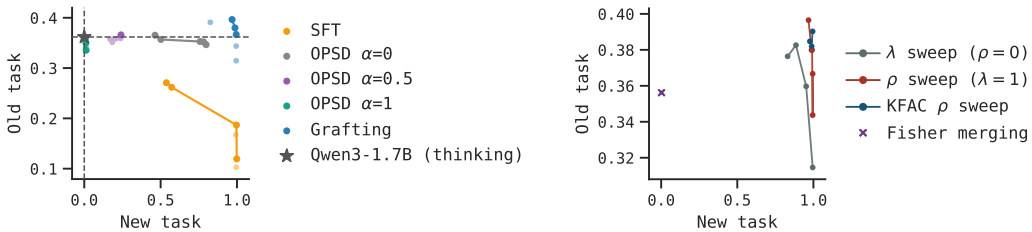


Figure 6: **(Left) Knowledge injection results. (Right) Disentangling the contribution of grafting's components.** The protection mask (ρ sweep) dominates no mask (λ sweep).

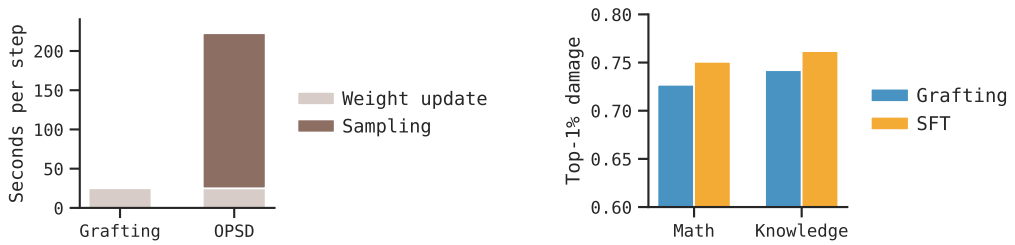


Figure 7: **(Left) Grafting is much faster to train than OPSD.** We show the per-step sampling and weight update time for Qwen3-8B (thinking). **(Right) Grafting writes less to sensitive coordinates than SFT.** We show the weighted delta $\sum_j F_{\text{ret}}(j)\Delta_j^2$ summed across the 1% of coordinates with the largest F_{ret}.

6 Analysis

Ablation studies Here we ablate the mask step that zeroes the most sensitive coordinates. Figure 6 (right) compares the protection mask (ρ sweep, in red) with plain model merging without the mask (λ sweep, in gray). We see that the protection mask Pareto-dominates plain model merging. Figure 6 (right) also shows Fisher merging (Matena & Raffel, 2022), which weights each coordinate by its Fisher ratio between the two tasks. We see that Fisher merging does not inject at all in this setting. We find that this is because the knowledge injection Fisher is usually dominated by the retention Fisher. Finally, the KFAC mask (§3, in navy) reaches the same old-task performance as the diagonal mask while zeroing far fewer directions: at ρ=0.001 it keeps new-task performance at 0.995 with old-task performance 0.390, which the diagonal mask needs ρ=0.01 to match. We note that KFAC being a better approximant to the retention Fisher does **not** directly imply interference must be more concentrated to few directions. The fact that retention of old tasks and injected knowledge can be achieved by zero-ing very few directions is in line with parallel work demonstrating the top few directions of curvature contain mostly interference (Zhu et al., 2026), and can be removed with little effect on task performance (Song et al., 2025).

Training latency Off-policy training also gets rid of the latency cost of on-policy sampling. We show in Figure 7 (left) the per-step wall clock time for both sampling and weight update for grafting and OPSD, both with Qwen3-8B (thinking). We see that sampling is roughly 90% of the step. In other words, grafting is $10\times$ faster to train than OPSD, while achieving much better performance.

Training on donor model writes less to sensitive weights We show in Figure 7 (right) the Fisher-weighted weight delta $\sum_j F_{\text{ret}}(j)\Delta_j^2$ that falls inside the top 1% of coordinates with the largest F_{ret}, normalized by that of all coordinates. In other words, this number shows how much a weight update writes into the most sensitive coordinates of the post-trained weights. We first see that the 1% coordinates explain nearly 75% of the weight changes, which empirically justifies the protection mask. More importantly, we see that grafting writes less to the sensitive weights than SFT, for both math and knowledge learning tasks, which helps explain why the protection mask alone cannot push SFT to the same frontier as grafting, as we see above. In Appendix G, we show how this effect can be predicted by a minimal linear model.

7 Related Work

Continual learning for post-trained models. Continually trained networks can overwrite earlier capabilities (McCloskey & Cohen, 1989). Classic remedies constrain the update written into the parameters, e.g., elastic weight consolidation (Kirkpatrick et al., 2017). For language models, it has been shown that low-rank updates (Hu et al., 2022) forget less but also learn less (Biderman et al., 2024). SFT and RL learn in different ways (Chu et al., 2025; Matsutani et al., 2025), and SFT in particular memorizes at the cost of generalization. This echoes the common belief is that continual training should be on-policy, and has been demonstrated by the success of continual training models using RL with verifiable rewards (RLVR) (DeepSeek-AI, 2025). Chen et al. (2025) attribute RL's smaller forgetting specifically to its use of on-policy data, rather than to other algorithmic choices such as KL regularization or advantage estimation. In practice, however, most data for continual learning is off-policy, such as knowledge after the pretraining cutoff (Jang et al., 2022), expert

demonstrations (Taori et al., 2023), delayed feedback from environments (Trajectory Labs, 2026), and synthetic data for self-improvement (Zelikman et al., 2022).

On-policy self-distillation and its failure modes. OPSD converts off-policy data into an on-policy learning signal, by distilling a solution-conditioned copy of the model into itself on its own samples (Zhao et al., 2026; Shenfeld et al., 2026; Hubotter et al., 2026; Penaloza et al., 2026). The approach is being scaled in practice (Trajectory Labs, 2026). Its benefit, however, appears to hold only within a limited scope of tasks. Reported failure modes include suppressed exploration, degeneration in reasoning length, and failure to generalize (Kim et al., 2026; Kaur et al., 2026). Recent work responds to these failures by filtering the part of the teacher signal that does not transfer to the student (Shen et al., 2026) or by routing token-level supervision between privileged teacher and student policies (Yu et al., 2026).

Model merging and weight-space operations. Weight-space operations are promised by loss-landscape geometry (Frankle et al., 2020; Neyshabur et al., 2020; Ainsworth et al., 2023), and a large literature does weight interpolation with models trained or finetuned with the same initialization (Wortsman et al., 2022a; Matena & Raffel, 2022; Li et al., 2022; Wortsman et al., 2022b; Ilharco et al., 2022), with refinements that resolve interference, linearize fine-tuning, or localize a skill in a sparse subset of the delta (Yadav et al., 2023; Yu et al., 2024; Ortiz-Jimenez et al., 2023; Panigrahi et al., 2023). Model merging has been shown to improve robustness (Wortsman et al., 2022b) and generation diversity (Dang et al., 2025). Most similar to our work, task arithmetic (Ilharco et al., 2023), also known as Shadow-FT (Wu et al., 2025) or chat vector (Huang et al., 2024), composes weight deltas trained on the base model. We improve over them with the proposed Fisher masking and donor model selection. Fisher merging (Matena & Raffel, 2022) derives a soft weighting between multiple tasks that train the model, while we derive a sparse mask using only the retention data and show that retention-only sparse masking is substantially more effective.

Models become harder to finetune. It has been known that training changes how well a model takes a further update, or plasticity (Ash & Adams, 2020; Dohare et al., 2024). Recent work demonstrates this for large language models. The longer a base model is pretrained beyond a certain point, the more sensitive its parameters become and the worse it finetunes (Springer et al., 2025). To mitigate such overtraining effect, it has been proposed to increase learning rate (Rofin et al., 2026), or more broadly, to bias pretraining toward flatter minima (Watts et al., 2026). Our experiments on varying the donor model for grafting point the same way: an earlier pretraining checkpoint can be a better θ_{donor} than the final base model and later post-trained intermediate checkpoints. Different from previous work, we show that the harder-to-finetune effect can be mitigated not only by training an easier-to-finetune model, but also by better continual update rules.

8 Conclusion and Future Work

We asked whether continual learning for RL-trained models has to be on-policy, and showed that it does not. Model merging alone already makes off-policy SFT indistinguishable from OPSD. Building on that, grafting obtains the weight delta by finetuning the donor model and masks the most destructive parts of the updates. Across three settings, two model families, and three sizes, grafting Pareto-dominates both SFT and OPSD in terms of in-domain injection and generalization. Which donor model to use for grafting turns out to be crucial too. We show that earlier pretraining checkpoints are usually a better choice, though future work can study more principled approaches to select the optimal pretraining checkpoint for grafting. Another direction to optimize for future work is to predict optimal λ and ρ instead of a sweep. More broadly, since off-policy data is what continual learning mostly has, we hope this work encourages more effort on making off-policy training generalize.

AI USE STATEMENT

In this work, we used generative AI tools for improving the writing of this paper, finding related work, and generating synthetic data, and implementing some code. We have not used generative AI tools for drafting section, proving claims. We have reviewed all AI-assisted work. We checked LLM-generated claims and code and retrieved related work and generated data. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stańczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, 2023.

Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. In *International Conference on Learning Representations (ICLR)*, 2023.

Jordan T. Ash and Ryan P. Adams. On warm-starting neural network training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John P. Cunningham. LoRA learns less and forgets less. *Transactions on Machine Learning Research*, 2024.

Souradip Chakraborty, Noah Ziems, Furong Huang, Meng Jiang, Amrit Singh Bedi, and Omar Khattab. Pedagogical RL: Teaching models to teach themselves from privileged information. 2026. https://noahziems.com/pedagogical-rl.

Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. Retaining by doing: The role of on-policy data in mitigating forgetting. *arXiv preprint arXiv:2510.18874*, 2025.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

Thomas M Cover, Joy A Thomas, and John Kieffer. *Elements of information theory*, volume 2. wiley New York, 1991.

Xingyu Dang, Christina Baek, Kaiyue Wen, J. Zico Kolter, and Aditi Raghunathan. Weight ensembling improves reasoning in language models. *arXiv preprint arXiv:2504.10478*, 2025.

DeepSeek-AI. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645:633–638, 2025.

DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Shibhansh Dohare, J. Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Rupam Mahmood, and Richard S. Sutton. Loss of plasticity in deep continual learning. *Nature*, 632:768–774, 2024.

Wei Du, Shubham Toshniwal, Branislav Kisacanin, Sadegh Mahdavi, Ivan Moshkov, George Armstrong, Stephen Ge, Edgar Minasyan, Feng Chen, and Igor Gitman. Nemotron-math: Efficient long-context distillation of mathematical reasoning from multi-mode supervision. *arXiv preprint arXiv:2512.15489*, 2025.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning (ICML)*, 2020.

Thomas George, César Laurent, Xavier Bouthillier, Nicolas Ballas, and Pascal Vincent. Fast approximate natural gradient descent in a kronecker factored eigenbasis. *Advances in neural information processing systems*, 31, 2018.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Shih-Cheng Huang, Pin-Zu Li, Yu-Chi Hsu, Kuang-Ming Chen, Yu Tung Lin, Shih-Kai Hsiao, Richard Tzong-Han Tsai, and Hung yi Lee. Chat vector: A simple approach to equip llms with instruction following and model alignment in new languages. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Jonas Hubotter, Frederike Lubeck, Lejs Deen Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Zarif Ikram, Arad Firouzkouhi, Stephen Tu, Mahdi Soltanolkotabi, and Paria Rashidinejad. Crispedit: Low-curvature projections for scalable non-destructive LLM editing. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=gTHlB3WK0l.

Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. Patching open-vocabulary models by interpolating weights. In *Advances in Neural Information Processing Systems*, 2022.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations (ICLR)*, 2023.

Joel Jang, Seonghyeon Ye, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, Stanley Jungkyu Choi, and Minjoon Seo. Towards continual knowledge learning of language models. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2110.03215.

Simran Kaur, Narutatsu Ri, Yinghui He, Liam Fowl, and Sanjeev Arora. Rethinking on-policy self-distillation for thinking models. *arXiv preprint arXiv:2607.05184*, 2026.

Jeonghye Kim, Xufang Luo, Minbeom Kim, Sangmook Lee, Dohyung Kim, Jiwon Jeon, Dongsheng Li, and Yuqing Yang. Why does self-distillation (sometimes) degrade the reasoning capability of llms? *arXiv preprint arXiv:2603.24472*, 2026.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Suhas Kotha, Jacob Mitchell Springer, and Aditi Raghunathan. Understanding catastrophic forgetting in language models via implicit inference. In *International Conference on Learning Representations*, 2024.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. Branch-train-merge: Embarrassingly parallel training of expert language models. *arXiv preprint arXiv:2208.03306*, 2022.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pp. 2408–2417. PMLR, 2015.

Michael Matena and Colin Raffel. Merging models with fisher-weighted averaging. In *Advances in Neural Information Processing Systems*, 2022.

Kohsei Matsutani, Shota Takashiro, Gouki Minegishi, Takeshi Kojima, Yusuke Iwasawa, and Yutaka Matsuo. Rl squeezes, sft expands: A comparative study of reasoning llms. *arXiv preprint arXiv:2509.21128*, 2025.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Academic Press, 1989.

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? In *Advances in Neural Information Processing Systems*, 2020.

NVIDIA. Nemotron-sft-math-v4. https://huggingface.co/datasets/nvidia/Nemotron-SFT-Math-v4, 2026. Hugging Face dataset.

OLMo Team. OLMo 3: Charting a path through the model flow. *arXiv preprint*, 2026.

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Abhishek Panigrahi, Nikunj Saunshi, Haoyu Zhao, and Sanjeev Arora. Task-specific skill localization in fine-tuned language models. In *International Conference on Machine Learning (ICML)*, 2023.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models. *arXiv preprint arXiv:2602.04942*, 2026.

Mark Rofin, Aditya Varre, and Nicolas Flammarion. (how) learning rates regulate catastrophic overtraining. *arXiv preprint arXiv:2604.13627*, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Zhanming Shen, Jintao Tong, Shaotian Yan, Chen Shen, Hao Chen, Wentao Ye, Xiaomeng Hu, Rui Miao, Haobo Wang, Junbo Zhao, Gang Chen, and Jieping Ye. Purified opsd: On-policy self-distillation without losing how to think. *arXiv preprint arXiv:2607.02234*, 2026.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

Minhak Song, Kwangjun Ahn, and Chulhee Yun. Does sgd really happen in tiny subspaces? In *International Conference on Learning Representations*, volume 2025, pp. 8086–8120, 2025.

Jacob Mitchell Springer, Sachin Goyal, Kaiyue Wen, Tanishq Kumar, Xiang Yue, Sadhika Malladi, Graham Neubig, and Aditi Raghunathan. Overtrained language models are harder to fine-tune. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2503.19206.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

Trajectory Labs. Scaling sdpo. https://trajectory.ai/field-notes/scaling-sdpo, 2026. Field notes; cited for delayed environment feedback as an off-policy data source.

Ishaan Watts, Catherine Li, Sachin Goyal, Jacob Mitchell Springer, and Aditi Raghunathan. Sharpness-aware pretraining mitigates catastrophic forgetting. *arXiv preprint arXiv:2605.02105*, 2026.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, 2022a.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo-Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022b.

Taiqiang Wu, Runming Yang, Jiayi Li, Pengfei Hu, Yik-Chung Wu, Ngai Wong, and Yujiu Yang. Shadow-ft: Tuning instruct model via training on paired base model. *arXiv preprint arXiv:2505.12716*, 2025.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *International Conference on Machine Learning (ICML)*, 2024.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Honglin Yu, Weinan Dai, Yuxuan Song, Xiang Wei, Haodong Zhou, Jingjing Liu, Wei Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yong-Xu Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Xinlei Yu, Gen Li, Qingyi Si, Guibin Zhang, Yuqi Xu, Congcong Wang, Shuai Dong, Kaiwen Tuo, Xiangyu Zeng, Kaituo Feng, Qunzhong Wang, Yang Shi, Xiaobin Hu, Xiangyu Yue, Jiaqi Wang, and Shuicheng Yan. Dopd: Dual on-policy distillation. *arXiv preprint arXiv:2606.30626*, 2026.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, pp. 15476–15488, 2022.

Charlie Zhang, Graham Neubig, and Xiang Yue. On the interplay of pre-training, mid-training, and rl on reasoning language models. In *Forty-third International Conference on Machine Learning*, 2026.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Kejian Zhu, Zhuoran Jin, Shangqing Tu, Hongbang Yuan, Yushi Bai, Kang Liu, Juanzi Li, and Jun Zhao. Sft conflicts, rl coexists: A theoretical and empirical analysis of multi-task learning for llms. *arXiv preprint arXiv:2608.03573*, 2026.

A DATA

Knowledge (synthetic closed world). To guarantee the facts are new, we build a closed fictional world rather than reuse public corpora: 200 invented entities and 1,125 atomic facts about them. Each fact is written as question-answer pairs in 8 distinct phrasings, giving 9,000 QA pairs, and new-task performance is measured on held-out phrasings never seen in training, so recall cannot be inflated by memorized surface form. For OPSD we additionally write 200 wiki-style articles that state the facts, used only for conditioning. The facts are absent from the post-trained model: closed-book exact-match is $\approx 0\%$ (instruct 0.09%, base 0.13%). Answers are scored by rule-based `<answer>` exact-match. All methods train on the same mixture of the 9,000 QA pairs and the 1,533 AoPS traces of the math corpus.

Math reasoning, Qwen3-1.7B (thinking). Training traces are competition math solutions sampled from the AoPS split of Nemotron-SFT-Math (NVIDIA, 2026). We mine the hard subset by sampling Qwen3-1.7B (thinking) 16 times per problem at temperature 1.0 and keeping only problems it fails on all 16 attempts, so the target skill is absent. After length-filtering and deduplication we keep 1,533 training traces, and hold out a disjoint set of 111 mined-hard problems to measure new-task performance.

Math reasoning, Qwen3-8B (thinking). At 8B we mine the same pool with a looser criterion (so we have a meaningful number of samples), keeping problems the post-trained model solves at most once in 16 attempts, which leaves 985 training samples. A disjoint set of 100 problems is held out to measure new-task performance.

Math reasoning, Qwen3-8B (non-thinking). The non-thinking experiment uses problems from DAPO (Yu et al., 2025). Hard problems are mined the same way as above, by sampling Qwen3-8B in non-thinking mode at temperature 1.0 and keeping only the problems it never solves, and the worked solutions for those problems are the off-policy data. This leaves 1,163 training samples, with a disjoint set of 200 problems held out to measure new-task performance.

B PEDAGOGICAL RL

Pedagogical RL (Chakraborty et al., 2026) improves a model in two stages: (1) it synthesizes reasoning traces from problems and their reference solutions, and (2) it finetunes the model on the synthesized traces with SFT. We keep stage (1) unchanged and replace SFT in stage (2) with grafting.

Notation Following Chakraborty et al. (2026), let x denote a problem, c its reference solution without reasoning, and τ the reasoning trace to be synthesized. Two copies of the model play different roles: a privileged teacher

$$\pi_{\theta_T}(\tau \mid x, c),$$

whose prompt contains the reference solution and instructs it to solve the problem in its own way rather than copy c, and a student

$$\pi_{\theta_S}(\tau \mid x),$$

which sees only the problem.

Teacher training The teacher is trained to write traces that are both correct and plausible under the student. For each position t of a trace, let the student's most likely next token be

$$a_t^{\max} = \arg\max_a \pi_{\theta_S}\left(a \mid x, \tau_{<t}\right),$$

and define the surprise gap

$$d_t = \log \frac{\pi_{\theta_S}\left(a_t^{\max} \mid x, \tau_{<t}\right)}{\pi_{\theta_S}\left(\tau_t \mid x, \tau_{<t}\right)},$$

which is zero when the trace follows the student's top choice and grows as the trace takes tokens the student finds unlikely. The spike-aware learnability score aggregates the gaps over the T tokens of

the trace,

$$G^{\theta_S}_{\text{spike}}(\tau \mid x) = \exp\left[-\frac{\lambda}{\beta}\log\left(\frac{1}{T}\sum_{t=1}^{T}\exp(\beta d_t)\right)\right],$$

where β interpolates between penalizing the mean gap ($\beta \to 0$) and the largest gap ($\beta \to \infty$), and λ sets the strength of the penalty. The pedagogical reward multiplies the learnability score with the correctness reward R,

$$r_{\text{ped}}(x, c, \tau) = R(x, c, \tau)\, G^{\theta_S}_{\text{spike}}(\tau \mid x),$$

and the teacher is trained with GRPO (Shao et al., 2024) to maximize

$$\max_{\theta_T} \mathbb{E}_{\tau \sim \pi_{\theta_T}(\cdot|x,c)}\left[r_{\text{ped}}(x, c, \tau)\right].$$

The student π_{θ_S} stays fixed during teacher training and only scores how learnable the teacher's traces are. We denote the trained teacher by $\tilde{\pi}_T$.

Trace synthesis For each problem and its reference solution, we sample reasoning traces

$$\tau \sim \tilde{\pi}_T(\cdot \mid x, c)$$

from the trained teacher. These traces are the training data for stage (2).

Problems and solutions The pairs $\{(x, c)\}$ are the 1,533 training samples of the 1.7B teacher-trace setting (§4.1), covering 628 unique problems: x is the problem and c is the written solution of the teacher trace with its reasoning removed. Each pair also carries the problem's final answer, which the correctness reward R checks. The synthesized pairs $\{(x, \tau)\}$ are the off-policy data for stage (2), where grafting replaces SFT.

Hyperparameters The teacher and the student are both initialized from Qwen3-1.7B in thinking mode, and the maximum sequence length is 18K tokens.

C Training

Grafting and direct SFT (math). Both fine-tune with identical optimization and differ only in the checkpoint updated (the earlier checkpoint for grafting, the post-trained model for direct SFT). The grid spans nine learning rates (5×10^{-7} to 5×10^{-5}) and five epochs, snapshotting after each epoch, at global batch 32 and sequence length 18K. We optimize with AdamW (β=(0.9, 0.95), weight decay 0.01) under a cosine schedule with warmup ratio 0.1 and gradient clipping at 1.0.

Grafting and direct SFT (knowledge). Full fine-tuning at learning rate 2×10^{-5} with a cosine schedule, warmup ratio 0.1, $\beta = (0.9, 0.95)$, weight decay 0.01, gradient clip 1.0, global batch 32, and sequence length 18K. The knowledge QA samples receive an empty-think prefix (`<think>\n\n</think>`), while the replay samples receive the standard `<think>reasoning_content</think>`.

On-policy self-distillation (OPSD). OPSD follows grafting and SFT in terms of training hyperparameters. Here are OPSD-specific details: the model samples its own response conditioned on a demonstration and is trained to match a frozen, demonstration-conditioned teacher under the divergence D_α, with rollout n=1 and top-k 100 for the teacher logits. We report $\alpha \in \{0, 0.5, 1\}$ for OPSD.

D Evaluation

Decoding. All math scores are averaged over 16 rollouts at temperature 1.0 and top-p 1.0, decoding up to 32K tokens in thinking mode and 16K in non-thinking mode. Knowledge injection is decoded greedily.

New task. New-task performance is measured on a held-out set the model was never trained on, as defined in §A.

Old task. Old-task performance is the mean over the same three out-of-domain competition-math suites, AIME 2024, AIME 2025 and HMMT 2025. The quantity measures generalization when the continual learning task is math and retention when it is knowledge.

E OLMo-3 EXPERIMENTS

Data. We mine hard problems for OLMo-3-7B (thinking) the same way as for Qwen3-1.7B (thinking): sampling OLMo-3-7B (thinking) 16 times per problem at temperature 1.0 and keeping only problems it fails on all 16 attempts. We hold out 100 problems i.i.d. to measure new-task performance, and train on every trace of the rest.

Checkpoints. On the post-training side we use the four checkpoints OLMo-3 releases: the base model, the SFT checkpoint, the DPO checkpoint, and OLMo-3-7B (thinking) itself. On the pretraining side we use stage-3 (see OLMo Team (2026) for details) steps 11,921, which is the released base model, 9,000, 6,000 and 3,000, together with stage-2 step 47,684, the last stage-2 checkpoint. The gaps from the released base model are 2,921, 5,921, 8,921 and 11,921 steps, which we write as 3k, 6k, 9k and 12k.

Training. We sweep over five learning rates (5×10^{-6} to 5×10^{-5}), for five epochs at global batch 32 and maximum length 18,432, in thinking mode.

Training stage-2 checkpoints on long outputs. Stage-2 base checkpoints (OLMo Team, 2026) set the max lengths as 8,192 with no RoPE scaling, so we copy stage-3's YaRN block onto them to finetune at 18,432 tokens to enable training stage-2 base checkpoints on long reasoning traces.

F FISHER ESTIMATION

We discuss the various derivation and implementation details of the retention Fisher (3). Recall that the Diagonal Fisher approximation leads to scores that weight the donor update by entry, and the resulting projection (4) reduces to a $\{0,1\}$ mask corresponding to the entries with the highest scores. In general, let us consider an orthonormal basis $U \in \mathbb{R}^{d_\theta \times d_\theta}$. As examples, the Diagonal Fisher approximation corresponds to the standard coordinate basis $U = [e_i]$. On the other hand, if U is the eigenbasis of $F_{\text{ret}} = U\Sigma U^\top$, $\Sigma = \text{diag}(\sigma_i)$ then we note the corresponding scores and projection:

$$\Delta^\top F_{\text{ret}} \Delta = (U^\top \Delta)^\top \Sigma (U^\top \Delta) = \sum_{i=1}^{d_\theta} \sigma_i \langle u_i, \Delta \rangle^2 .$$

Thus, defining the score accordingly $s_i := \sigma_i \langle u_i, \Delta \rangle^2$ and the set $S_\rho := \{u_i : s_i \text{ in top-}\rho \text{ fraction}\}$, a mask in the diagonal case turns into a *projection* onto the orthogonal complement of S_ρ: $\mathcal{M}_\rho := \text{Proj}_{S_\rho}^{\perp}(\cdot)$. However, we note that the projection based on S_ρ is exact only when U diagonalizes F_{ret}, as otherwise projecting based on scores s_i **ignores the contribution from off-diagonal elements** $(F_{\text{ret}})_{ij}\Delta_i\Delta_j$, which can be greatly suboptimal if Δ_i and Δ_j are tightly coupled, e.g. parameters in the same weight matrix. The coordinate basis is often a suboptimal choice for this purpose, and an exact eigendecomposition of the full-dimensional Fisher is infeasible.

This is a similar motivation as prior optimization literature that seeks to approximate Natural Gradient Descent for parametric maximum likelihood estimation, which uses the Fisher Information matrix as a preconditioner. A seminal work on this front is the KFAC optimizer (Martens & Grosse, 2015). The essential observation is that the Fisher Information matrix can be approximated by a block-diagonal matrix, where each block corresponds to a layer's parameters; then, for matrix-shaped weights $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ this can further be approximated by a *Kronecker factor* between two symmetric matrices of shape $P \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$, $Q \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$. For example, given a feedforward weight matrix $h = Wa$, we have

$$P = \mathbb{E}_z \mathbb{E}_{y \sim p_\theta(\cdot \mid z)} \left[\frac{\partial L(\theta; y, z)}{\partial h} \frac{\partial L(\theta; y, z)}{\partial h}^\top \right], \quad Q = \mathbb{E}_z[aa^\top], \tag{5}$$

where $L(\theta; y, z) = -\log p_\theta(y|z)$ is the negative log-likelihood of the model's own generation y given input z. We note that for $<$ 2-D parameters the original Diagonal Fisher approximation

is made and scores computed as described in Section 3. Consider the eigendecompositions $P = U\Sigma U^\top$, $Q = VDV^\top$, and for a given update Δ, consider the (appropriately re-shaped) subset corresponding to the current layer Δ^W. Observe by the Kronecker mixed-product property we have:

$$\begin{aligned}\mathrm{vec}(\Delta^W)^\top (Q \otimes P)\mathrm{vec}(\Delta^W) &= \mathrm{Tr}(P\Delta^W Q \Delta^W) \\ &= \mathrm{Tr}(U\Sigma U^\top \Delta^W V D V^\top \Delta^W) \\ &= \mathrm{Tr}(\Sigma \tilde{\Delta}^W D (\tilde{\Delta}^W)^\top), \quad \tilde{\Delta}^W := U^\top \Delta^W V \\ &= \sum_{i,j=1}^{d_{\text{out}}, d_{\text{in}}} \sigma_i d_j (\tilde{\Delta}^W_{i,j})^2.\end{aligned}$$

Therefore, for this $d_{\text{out}} \cdot d_{\text{in}}$ parameter subset of Δ, the scores become $s_{(i,j)} = \sigma_i d_j (u_i^\top \Delta^W v_j)^2$. Tallying the scores up across all scalar/vector parameters and tensor parameters, the KFAC-based projection mask thus becomes a projection onto the orthogonal complement of $\{e_k\} \cup \{u_i \otimes v_j\}$, where $s_k, s_{(i,j)}$ belong to the top-ρ of scores across scalar/vector and matrix parameters.

Though a KFAC decomposition does not in general perfectly diagonalize the full Fisher Information, as it does not capture block-to-block dependencies, it is known empirically to be a significantly better decomposition than the diagonal (George et al., 2018).

Implementation. To estimate the retention Fisher matrix, we compute the per-token-normalized from 1858 rollouts of the Qwen3-1.7B thinking model θ_{post} at temperature 1.0 prompted with AoPS-style math problems. The ρ score threshold is estimated by taking a ρ-quantile of a subsample of the list of scores and using that as the selection threshold.

The Diagonal Fisher projection is implemented efficiently via a mask. The KFAC projection is slightly more arithmetic-intensive to compute due to the layer-wise eigendecompositions and resulting matrix multiplications, but remains efficient due to the layer-wise factorization.

G A MINIMAL MODEL FOR WHY GRAFTING FORGETS LESS

Why does SFT interfere more with post-trained capabilities than grafting? We argue that models trained on narrow distributions (e.g., post-training and late stages of pretraining) explain new data differently, also known as *implicit task inference* (Kotha et al., 2024). We demonstrate this intuition with a minimal model.

Consider two linear regression tasks with data distributions $\mathcal{D}_A$ and $\mathcal{D}_B$. For each task $T \in \{A, B\}$,

$$x \sim \mathcal{D}_T, \qquad y = \beta_T^\top x, \tag{6}$$

Task A is a post-trained capability we want to retain, while task B is a capability we want to continually learn. We model the two tasks with a mixture of linear experts,

$$\hat{y}(x) = g_\phi(x) w_A^\top x + (1 - g_\phi(x)) w_B^\top x, \qquad g_\phi(x) = \sigma(\phi^\top x). \tag{7}$$

We assume pretraining learns both tasks imperfectly but learns their assignment relatively well. On task-B inputs, assume for simplicity that $g_{\text{pre}}(x) = \varepsilon_D, x \sim \mathcal{D}_B$. After post-training on task A, the gate becomes more biased toward expert A on the same task-B inputs $g_{\text{post}}(x) = \varepsilon_P, x \sim \mathcal{D}_B$, with $0 \le \varepsilon_D < \varepsilon_P < 1$, which is shown by Kotha et al. (2024). Thus ε measures how much task-B data is routed through expert A.

We now analyze fine-tuning on task B. For a checkpoint with task-B assignment ε, under squared loss,

$$L_B = \frac{1}{2} r_B^\top \Sigma_B r_B, \quad \text{where} \tag{8}$$

$$r_B = \varepsilon w_A + (1 - \varepsilon) w_B - \beta_B, \tag{9}$$

$$\Sigma_B = \mathbb{E}_{x \sim \mathcal{D}_B}[x x^\top]. \tag{10}$$

The gradients with respect to the two experts are

$$\nabla_{w_A} L_B = \varepsilon \Sigma_B r_B, \qquad \nabla_{w_B} L_B = (1-\varepsilon)\Sigma_B r_B. \tag{11}$$

Therefore the same task-B learning signal is routed to the two experts in the ratio

$$\frac{\|\nabla_{w_A} L_B\|}{\|\nabla_{w_B} L_B\|} = \frac{\varepsilon}{1-\varepsilon}. \tag{12}$$

Hence task assignment directly controls how much fine-tuning on task B modifies the expert supporting task A. Since we assume $\varepsilon_D < \varepsilon_P$, finetuning the pretrained model routes less of the task-B learning signal through task A than directly fine-tuning the post-trained model. This is aligned with what we show in §6.