REJUVENATING MODEL PLASTICITY FOR ROBUST SFT-TO-RL HANDOFF

Anonymous authors
Paper under double-blind review

ABSTRACT

Supervised Fine-Tuning (SFT) followed by Reinforcement Learning (RL) has become a standard pipeline for Large Language Model (LLM) post-training. SFT is expected to provide a useful behavioral prior for RL to further enhance model capabilities. However, checkpoints with excessive SFT often show limited improvement during RL. We attribute this failure to the loss of model plasticity: the reduced ability of an SFT-initialized policy to be effectively reshaped by subsequent RL. To better understand this phenomenon, we conduct detailed analysis from multiple perspectives, including parameter changes, output spaces, and RL optimization dynamics. Our results show that models from excessive SFT tend to produce over-confident token distributions and exhibit sharp parameter landscapes, which make them harder to optimize in the RL stage. To enable a more robust SFT-to-RL handoff, we propose `Rejuvenation`, a simple yet effective method that restores plasticity while preserving useful SFT-acquired priors. Rejuvenation leverages base-anchored model fusion to reduce excessive SFT-induced drift with targeted neuron reset to mitigate model rigidity. Experimental results on both math reasoning tasks and agentic tasks demonstrate that our approach recovers RL performance from checkpoints at different SFT stages, widening the viable SFT-to-RL handoff window.

1 INTRODUCTION

Post-training has emerged as a critical phase for unlocking the reasoning and agentic capabilities of large language models (LLMs) (OpenAI, 2024; Guo et al., 2025). Many practical systems adopt the SFT-then-RL (Guo et al., 2025) pipeline, where supervised fine-tuning (SFT) (Ouyang et al., 2022; Bai et al., 2022) first teaches models to follow instructions, produce the desired format, and acquire cold-start knowledge, after which reinforcement learning (RL) (Shao et al., 2024; Zhang et al., 2025a) further optimizes the policy according to task rewards (Yue et al., 2025; Liu et al., 2025a). In this pipeline, SFT is not merely for imitating high-quality solutions, but also determines the initialization that RL inherits. This pipeline therefore relies on an implicit assumption: after acquiring useful behaviors through imitation, the SFT checkpoint should still serve as a suitable starting point for reward-driven optimization.

However, the amount of SFT needed for strong supervised behavior may not coincide with the amount of SFT that yields the best final model after RL. Intuitively, if SFT is stopped too early, the model may be under-prepared for RL, lacking the instruction-following patterns or task-specific behaviors needed for efficient optimization. Meanwhile, if SFT is continued for too long, the model may become overly specialized to the supervised data, and the resulting checkpoint may leave limited room for RL to further improve the policy and generalize (Chu et al., 2025). Thus, despite its widespread success, the handoff from SFT to RL remains a fragile and computationally expensive. A critical yet often overlooked challenge is to determine which SFT checkpoint should be used as the RL initial policy.

Standard SFT metrics (e.g., training loss, validation accuracy) measure imitation quality, not the checkpoint's capacity for reward-driven improvement. A natural remedy is early stopping (Li et al., 2026), but it still assumes that the appropriate stopping point can be reliably identified, and therefore does not resolve the core problem. Recent work improves the SFT stage through better data (Huang et al., 2026; Zhao et al., 2026), objectives (Fu et al., 2026; Zhu et al., 2026b), or training recipes (Zhang et al., 2025b), but these methods primarily change the SFT trajectory rather than provide a reliable criterion for RL readiness. As a result, practitioners still often have to launch RL

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.

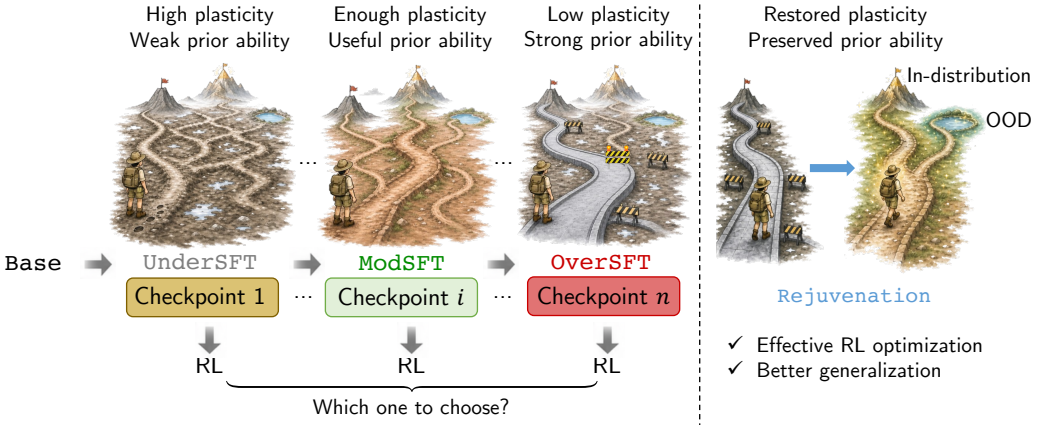


Figure 1: Overview of Rejuvenation.

from multiple SFT checkpoints to determine the appropriate handoff point. This search is expensive, and its outcome can be sensitive to RL hyperparameters and optimization noise.

In this paper, we aim to analyze and address this critical dilemma in the SFT-to-RL handoff, where insufficient SFT may prematurely stop imitation learning that prevents the model from acquiring useful skills, while over-trained SFT becomes highly resistant to further improvement via RL. To better understand this issue, we conduct a detailed analysis of how extended SFT changes both model behavior and subsequent optimization dynamics under RL. We find that OverSFT models tend to produce sharper, more over-confident output distributions and less smooth parameter space, showing large gradient norms but limited performance gains and smaller parameter update magnitude compared to ModSFT models. These findings suggest that the excessive SFT does not merely overfit the supervised data, but also make the policy resistant and becomes less effectively adaptable in the subsequent RL stage. We identify this failure mode as a loss of *model plasticity*[1] (Dohare et al., 2024; Han et al., 2026): the model becomes difficult to reshape through RL.

Based on this observation, we propose Rejuvenation, a simple yet effective post-hoc mechanism that enables robust SFT-to-RL handoff for recovering model plasticity, which avoids complex SFT loss modifications and costly checkpoint searches. Our key insight is that SFT should provide a useful behavioral prior, but not at the cost of losing plasticity. Our approach is a dual-level strategy. First, at the global level, we utilize base-anchored model fusion to reduce excessive SFT-induced drift while retaining useful behavior. At the local level, we introduce a targeted neuron reset mechanism based on logit attribution to selectively restore diversity in over-confident LLM most responsible for collapsed predictions. As shown in Figure 1, our method effectively mitigates the resulting rigidity induced by excessive SFT, while preserving the effective behavioral prior acquired from sufficient SFT. We evaluate Rejuvenation on both mathematical reasoning tasks and agentic tasks. Experiments show that our method consistently recovers RL performance from OverSFT models and brings checkpoints from different SFT stages to an overall performance level comparable to ModSFT+RL, widening the viable handoff window.

The main contributions of this work can be summarized as follows:

1. We identify a failure mode in the SFT-then-RL pipeline: the SFT-to-RL handoff dilemma, where fully-trained SFT models lose plasticity and limit RL improvement.
2. We provide detailed analysis from multiple perspectives, revealing that over-SFT leads to reduced effective gradients during RL, which leads to entropy collapse and fundamentally hurts RL optimization dynamics.
3. We propose an effective rejuvenation method to recover model plasticity post-hoc using model fusion and neuron reset, making it robust to different SFT-to-RL handoffs.
4. We demonstrate the effectiveness of our method on both mathematical and agentic tasks, showing that it recovers RL performance across different SFT durations and makes the SFT-to-RL handoff less sensitive to checkpoint selection.

[1]We use model plasticity to refer to the ability of SFT models to undergo reward-driven improvement in subsequent RL. A plastic model should remain responsive to RL update and such updates can effectively translate into task performance gains.

2 Related Work

SFT and RL in LLM Post-Training. RL has become an effective phase of LLM post-training (OpenAI, 2024; Guo et al., 2025; Shao et al., 2024; Yu et al., 2025). Existing work improves the interaction between SFT and RL through off-policy data (Yan et al., 2025; Chen et al., 2025; Liu et al., 2025c; Ma et al., 2026; Huang et al., 2025), unified objectives (Liu et al., 2025b; Fu et al., 2026; Zhang et al., 2026b; Lv et al., 2025; Gan et al., 2026), and importance-weighted SFT losses (Zhu et al., 2026b; Qin & Springenberg, 2025; Zhang et al., 2026a). Carefully controlled SFT-then-RL nevertheless remains a strong baseline, and its outcome can be sensitive to SFT implementation details (Limozin et al., 2026). Other studies show that SFT and RL can have different generalization behavior (Chu et al., 2025), that RL may heal some SFT-induced forgetting only within a suitable checkpoint range (Jin et al., 2025b;a), and that high SFT scores do not reliably predict post-RL performance (Kang et al., 2026; Zhang et al., 2026a). These findings motivate checkpoint selection and objective redesign, whereas we study what can be done after selection has already failed.

Preventive SFT Regularization versus Post-Hoc Recovery. Many methods aim to prevent overfitting or excessive policy drift during SFT. GEM (Li et al., 2025), PSFT (Zhu et al., 2026b), ASFT (Zhu et al., 2026a), and CurioSFT (Wang et al., 2026) add regularization to maintain diversity, while DFT (Wu et al., 2026), AESL (Li et al., 2026), and ProFit (Liu et al., 2026) reweight the supervised objective. These methods alter the SFT trajectory from the outset and are complementary when retraining is possible. `Rejuvenation` addresses a different intervention point: it assumes the late SFT checkpoint already exists and repairs that checkpoint for subsequent RL.

3 Diagnosing and Restoring Plasticity in Over-Trained Models

In this section, we analyze why over-trained SFT models become difficult to improve with RL and then introduce two post-hoc recovery operations. We first investigate what excessive SFT changes in both parameter and output spaces before any RL update is applied, to understand the handoff-specific question: *does continued SFT move the checkpoint into a state that is less amemnable to subsequent RL optimization?* We then connect these changes to poor RL trainability, where large gradient norms do not translate into effective parameter movement or meaningful performance gains (Section 3.2). Motivated by these observations, we recover plasticity at two levels: base-anchored model fusion globally pulls the model toward a smoother region (Section 3.3), while attribution-guided neuron reset locally restores the high-contribution directions responsible for abnormal logits (Section 3.4).

3.1 How Does Over-Training Change the SFT Model?

3.1.1 Parameter Space

We train EvoLM-4B (Qi et al., 2025) on the math SFT data and save checkpoints along the SFT process. We denote the moderately trained checkpoint (epoch=2) as `ModSFT` and the over-trained checkpoint (epoch=32) as `OverSFT`. More training details are provided in Section 4.1 and Appendix B.

Excessive SFT induces large parameter shifts and weight magnitude. We first examine how SFT changes the model parameters. For each checkpoint, we visualize the element-wise parameter difference with respect to the base model. As shown in Figure 2 and the first row of Figure 3, we observe that `ModSFT` only introduces moderate and relatively smooth parameter changes, while `OverSFT` leads to extremely large parameter shifts and there are sharp spikes in the shifts, resulting in larger weight magnitude. These spikes indicate that over-training does not simply continue improving the same solution found by moderate SFT. Instead, it drives a small subset of parameters significantly far away from the base model. Additionally, these observations are consistent across all modules and layers. Please find Appendix C for more visualizations.

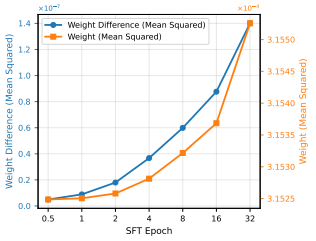


Figure 2: Parameter changes and statistics of different SFT checkpoints.

Subsequent RL induces much smaller movement than the preceding SFT-induced drift. We further examine how RL changes the model parameters by plotting the difference between: (1) RL vs SFT and (2) RL vs Base. Figure 3 shows that the subsequent RL stage only introduces much smaller changes compared with the SFT-induced shift. This suggests that once SFT has caused relatively high parameter distortion, RL is unlikely to reverse it through standard policy optimization.

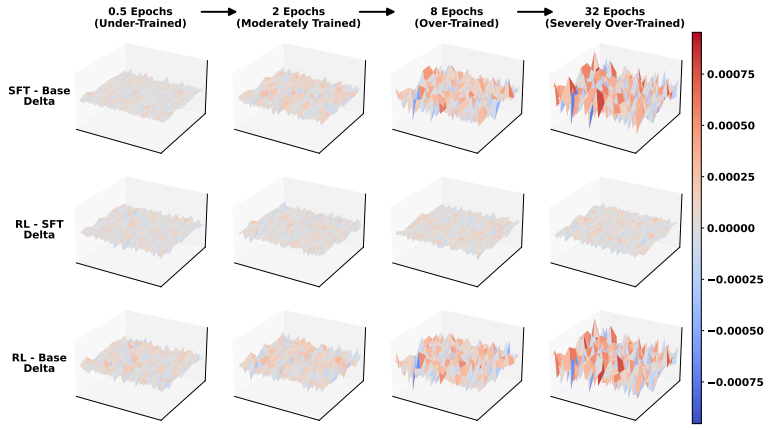


Figure 3: Parameter changes of `layers.0.self_attn.v_proj.weight` induced by SFT and RL.

3.1.2 Output Space

Excessive SFT saturates the policy before RL. We compare the output diversity of different SFT checkpoints in Table 1. We can observe that `OverSFT` achieves near-zero training loss but exhibits substantially lower token entropy than `ModSFT` (0.024 vs. 0.184), suggesting that excessive SFT drives the policy toward an over-confident regime. Although `OverSFT` attains a higher Pass@1 score than `ModSFT`, its Pass@K-Pass@1 gap is the smallest, indicating reduced output diversity despite stronger greedy performance. Consistently, Figure 7 shows logit distributions where `OverSFT` concentrates probability mass almost entirely on a single token (Li et al., 2025), whereas `ModSFT` still assigns non-negligible probability to alternative tokens.

Table 1: Diversity-related metrics of different SFT checkpoints.

Checkpoint	Training Loss	Entropy	MATH-500 Pass@1	Pass@K - Pass@1 Gap
`UnderSFT`	0.178	0.281	9.7	13.7
`ModSFT`	0.111	0.184	15.2	**15.3**
`OverSFT`	0.002	0.024	**15.8**	11.2

Takeaway

Over-trained SFT models differ from moderately trained models in both parameter and output spaces: they contain large parameter shifts and produce over-confident token distributions. These two signatures indicate reduced plasticity before RL starts.

3.2 Does RL Fail after Over-Trained SFT?

3.2.1 Training Dynamics

Over-trained models exhibit larger RL gradient norms. We next study whether these SFT-induced changes affect RL optimization. Figure 4 shows an apparent paradox: `OverSFT` has a much larger gradient norm than `ModSFT` throughout RL, yet its RL-induced parameter shift is smaller (Figure 3, RL-vs-SFT row) and its reward improvement is smaller as well.

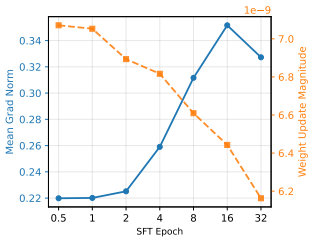


Figure 4: Average gradient norm and weight update magnitude of RL with different SFT checkpoints.

3.2.2 RL Performance Gain

Over-training reduces RL performance gain. Figure 5 shows that after excessive SFT training, the RL gain of `OverSFT` models drop significantly compared with `ModSFT` models, demonstrating that over-training harms the RL performance gain.

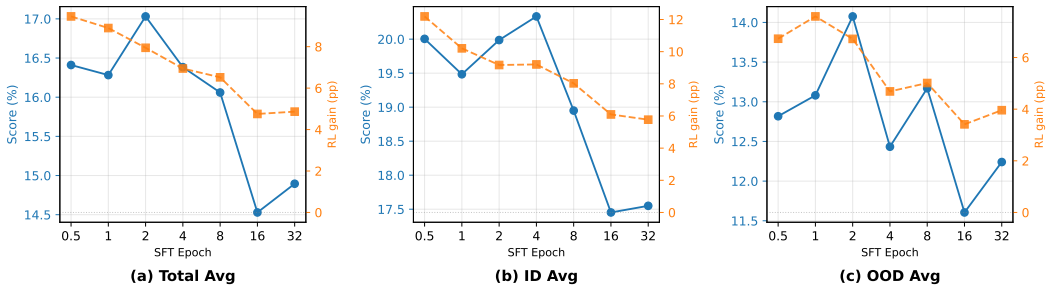


Figure 5: Evaluation results of RL-trained models from different SFT checkpoints. (a) Average score of ID and OOD tasks. (b) Average score on ID tasks. (c) Average score on OOD tasks.

Takeaways

- Over-trained models are hard to optimize during RL, exhibiting higher gradient norm but inducing less weight update magnitude.
- Over-training leads to smaller performance gains after RL on both ID and OOD tasks.

3.3 Global Recovery via Base-Anchored Fusion

Section 3.1 shows that over-training pushes a small set of parameters far from the base model and places the model in a sharp region. A natural fix is to pull the over-trained model back toward the base so that the shift shrinks and the surrounding landscape becomes smoother, while still keeping the useful behavior that SFT has acquired (Wortsman et al., 2022; Ilharco et al., 2023).

Base-anchored linear fusion. We therefore perform a simple element-wise linear interpolation between the base model and the `OverSFT` model:

$$\theta_{\text{fuse}} = \alpha\,\theta_{\texttt{OverSFT}} + (1-\alpha)\,\theta_{\texttt{Base}}, \qquad \alpha \in [0,1]. \tag{1}$$

The interpolation is applied to all model parameters, including attention and MLP weights as well as RMSNorm weights, the final norm, and `lm_head`. The fusion weight α controls how much of `OverSFT` is retained: $\alpha \to 1$ reduces to the original `OverSFT` model, and $\alpha \to 0$ falls back to the base model.

Why fusion helps. This operation scales the SFT-induced weight delta $\theta_{\texttt{OverSFT}} - \theta_{\texttt{Base}}$ by α. From the task-vector perspective of model editing and composition (Ilharco et al., 2023), the SFT update $\theta_{\texttt{OverSFT}} - \theta_{\texttt{Base}}$ can be viewed as a task vector that moves the base model toward the SFT solution. Base-anchored fusion preserves the direction of this task vector while reducing its magnitude, thereby retaining much of the task-specific behavior learned during SFT but avoiding the full parameter displacement caused by over-SFT. Additionally, fusion shrinks the element-wise deviation from the base to $\alpha \cdot (\theta_{\texttt{OverSFT}} - \theta_{\texttt{Base}})$, which reduces the large shifts and sharp spikes in Figure 3. The fused model therefore sits in a smoother region that is easier for RL to optimize. In practice, a moderate α (e.g., $\alpha = 0.5$) recovers most of the training-dynamics signals and token entropy while preserving much of the ID performance gained from SFT.

3.4 Local Recovery via Attribution-Guided Neuron Reset

Global fusion treats every parameter equally and cannot pinpoint which parameters actually drive the over-confident behavior. In particular, only a small number of neurons account for most of the over-confident logits, and scaling all parameters uniformly toward the base cannot selectively relax these neurons without weakening the rest. We therefore add a targeted step that first locates the neurons most responsible for the collapsed token distributions via direct logit attribution, a residual-stream decomposition technique commonly used in Transformer circuit analysis (Elhage et al., 2021), and then resets these neurons in θ_{fuse} back to their base-model values.

3.4.1 Identifying Abnormal Logit Contributors

Step 1: Selecting over-confident target tokens. Over-confidence concentrates on a subset of response tokens rather than being spread uniformly, so we first find where the effect is strongest and then attribute the logits only at those positions. Given a calibration set of prompt–response pairs, we run both the base model and `OverSFT` in teacher-forcing mode and record the per-position token

entropy $H_{\texttt{Base}}(t)$ and $H_{\texttt{OverSFT}}(t)$. For each sample, we pick the Top-N response positions with the largest entropy gap $|H_{\texttt{OverSFT}}(t) - H_{\texttt{Base}}(t)|$ as the target tokens $\mathcal{T}$. These are the positions where OverSFT has most aggressively sharpened the distribution relative to the base model, making them the most informative anchors for attribution.

Step 2: Decomposing logits into per-neuron contributions. Under a frozen final-RMSNorm approximation, the target logit on OverSFT can be written as a sum of residual-stream contributions, projected onto a fixed direction that depends on the target token y^*:

$$\begin{aligned} \tilde{z}_{y^*}(t) &= b_{y^*} + \langle e(t), d_{y^*}(t)\rangle + \sum_{\ell}\Big[\langle r_\ell^{\text{attn}}(t), d_{y^*}(t)\rangle + \langle r_\ell^{\text{mlp}}(t), d_{y^*}(t)\rangle\Big], \\ d_{y^*}(t) &= s_{\text{RMS}}(t)\cdot\big(\gamma \odot W_U[y^*]\big), \end{aligned} \tag{2}$$

where b_{y^*} is the optional lm_head bias for y^*[2], $e(t)$ is the token-embedding write, $r_\ell^{\text{attn}}(t)$ and $r_\ell^{\text{mlp}}(t)$ are the residual-stream writes from the attention and MLP blocks at layer ℓ, γ is the gain of the final RMSNorm, $s_{\text{RMS}}(t)$ is its normalization scalar at position t, and $W_U[y^*]$ is the unembedding row of y^*. This decomposition becomes exact only because we freeze $s_{\text{RMS}}(t)$ at its forward-pass value and replace the nonlinear RMSNorm with the linear projection $d_{y^*}(t)$. In our implementation we therefore track the reconstruction error $|\tilde{z}_{y^*}(t) - z_{y^*}(t)|$ to confirm that this approximation is tight. We use the resulting scores as attribution signals for ranking rather than as a complete causal explanation, since direct logit attribution can be misleading when later components erase or overwrite earlier residual-stream directions (Janiak et al., 2024).

For the modules that directly write to the residual stream, namely o_proj and down_proj, equation equation 2 can be further pushed down to the input-neuron level, giving an exact per-neuron contribution

$$s_i^{(\ell,m)} = a_i^{(\ell,m)}(t)\cdot\big\langle W_{:,i}^{(\ell,m)},\ d_{y^*}(t)\big\rangle, \qquad m \in \{\texttt{o_proj}, \texttt{down_proj}\}, \tag{3}$$

where $a_i^{(\ell,m)}(t)$ is the neuron activation at position t and $W_{:,i}^{(\ell,m)}$ is the corresponding column of the writer weight.

For the remaining internal projections (q_proj, k_proj, v_proj, up_proj, gate_proj), whose outputs do not directly write to the residual stream, no such exact neuron-level decomposition is available. We therefore score them with a local "gradient $\times$ activation" proxy on the same target logit $z_{y^*}(t)$. This proxy can be viewed as a first-order Taylor approximation to the change in the target logit induced by perturbing or removing the corresponding activation, and has been commonly used as a gradient-based importance estimate for neural units and Transformer components (Molchanov et al., 2017; Michel et al., 2019). Specifically, for k_proj and v_proj the score is aggregated over all prefix positions $0..t$, since these projections are read by attention at every later step. For q_proj, up_proj, and gate_proj, it is taken at the target position only. For q_proj and k_proj, the implementation uses pre-RoPE projection outputs, so these scores should be read as local attribution proxies rather than exact causal decompositions. This gives a uniform per-neuron score $s_i^{(\ell,m)}$ across all linear modules of the network, which we use for ranking in Section 3.4. The visualization of logit attribution is shown in Figure 8 in Appendix C.

3.4.2 Resetting high-attribution neurons

Building the reset set. For each selected target token t, we rank all neurons across all replaceable modules by their most positively contributing to the target logit and keep the top ρ fraction, yielding a token-level set $\mathcal{S}_{x,t}$ for each prompt x. The final reset set is the union over all selected tokens and calibration examples:

$$\mathcal{S} = \bigcup_{(x,t)\in\mathcal{T}} \mathcal{S}_{x,t}. \tag{4}$$

In our implementation, ρ controls the per-token selection budget; we report the resulting effective whole-model reset ratio after taking the union. We introduce a token-level gating variable $\omega^{(i)} = \mathbb{I}(i \in \mathcal{S}) \in \{0, 1\}$ based on the above definition.

[2] b_{y^*} denotes the lm_head bias for token y^* if present (e.g., models without tied embeddings); for models with bias-free lm_head (as in Llama (Grattafiori et al., 2024) and Qwen (Yang et al., 2024; 2025)), this term vanishes.

Table 2: Evaluation results on ID and OOD benchmarks of EvoLM-4B. The highest values before RL are underlined and the highest values after RL are **bolded**.

Method	GSM8K	MATH	AMC23	Miner.	Olymp.	ID Avg.	GPQA	ARC	MMLU	OOD Avg.	Avg.
ModSFT	28.1	15.2	4.3	4.0	2.4	10.8	5.3	13.4	3.4	7.4	9.1
+RL	**51.6**	25.7	6.4	11.6	**4.7**	**20.0**	**13.2**	22.9	6.2	14.1	17.0 (+7.9)
OverSFT	31.7	15.8	3.8	5.1	2.4	11.8	7.8	13.7	3.3	8.3	10.0
+RL	45.2	22.0	6.6	9.9	4.1	17.5	9.9	21.3	5.6	12.2	14.9 (+4.9)
DFT	30.9	14.5	5.3	5.2	2.3	11.7	5.8	7.2	3.3	5.4	8.5
+RL	43.6	24.4	7.5	**11.9**	3.9	18.3	8.3	22.3	5.7	12.1	15.2 (+6.7)
Rejuvenation	18.5	12.0	3.0	4.3	2.5	8.1	3.6	2.6	2.1	2.8	5.4
+RL	49.1	**26.2**	**8.0**	10.8	4.2	19.7	12.9	**26.2**	**6.9**	**15.3**	**17.5 (+12.1)**

Reset operation. Inspired by attribution-based neuron intervention in pretrained Transformers (Dai et al., 2022), we then overwrite each selected neuron in θ_{fuse} with its base-model value while leaving the rest untouched:

$$\theta^{(i)}_{\text{reset}} = \omega^{(i)}\theta^{(i)}_{\texttt{Base}} + (1-\omega^{(i)})\theta^{(i)}_{\text{fuse}} \tag{5}$$

where each neuron index corresponds to either a row or a column of the underlying weight matrix, depending on the module's role in the residual stream: rows for `q_proj`, `k_proj`, `v_proj`, `up_proj`, `gate_proj`, and columns for `o_proj` and `down_proj`. The full procedure is summarized in Algorithm 1.

Why reset helps. Reset and fusion play complementary roles. Fusion smooths the parameter space globally and brings the model into a better-conditioned region, but leaves every direction scaled uniformly, so the neurons that dominate the over-confident logits remain close to their over-trained configuration up to a factor of α. The reset step replaces exactly these rigid directions with base-model values, breaking the shortcut that produces collapsed token distributions. Because $|\mathcal{S}|$ is very small, the vast majority of SFT-acquired behavior in θ_{fuse} is preserved, while diversity is restored precisely where it was most lost. As shown in Figure 10, the output entropy on the affected tokens recovers substantially after the reset, and the resulting model becomes more responsive to subsequent RL optimization.

4 Experiments

4.1 Setup

Models and Baselines. We adopt EvoLM-4B (Qi et al., 2025) as the base model for mathematical tasks and Qwen3-8B (Yang et al., 2025) for agentic tasks. We compare Rejuvenation against ModSFT (epoch=2), OverSFT (epoch=32), and DFT (Wu et al., 2026), and report the corresponding +RL results under the same RL recipe. Detailed baseline descriptions are provided in Appendix B.

Training Details. For SFT training, we use LlamaFactory (Zheng et al., 2024b) for EvoLM-4B and slime (Zhu et al., 2025) for Qwen3-8B, both with the AdamW (Kingma, 2014) optimizer and a learning rate of 3.0×10^{-6}. For RL training, we use verl (Sheng et al., 2025) for mathematical tasks and slime for agentic tasks, both with GRPO (Shao et al., 2024) and a learning rate of 1×10^{-6}. For each checkpoint, we compare direct RL with applying Rejuvenation before the same RL training.

Evaluation Details. We evaluate on five ID mathematical benchmarks, τ-bench Retail and Airline for agentic tasks, and three OOD benchmarks. We report Pass@1 for mathematical and general benchmarks and task success rates for agentic tasks. Avg. is the mean of the ID and OOD averages. The full benchmark list, decoding parameters, and training hyper-parameters are provided in Appendix B.

4.2 Can a Missed Handoff Be Repaired?

We first study whether Rejuvenation can repair a missed handoff from the 32-epoch OverSFT checkpoint, using ModSFT+RL (epoch=2) as a reference under the same RL recipe.

Math. Table 2 reports the evaluation results on EvoLM-4B and we have three observations: (1) Starting RL from OverSFT leads to a clear performance drop compared with ModSFT (17.5 vs. 20.0 ID Avg. and 12.2 vs. 14.1 OOD Avg.). (2) Replacing SFT with DFT does not fully recover the

Table 3: Evaluation results on ID and OOD benchmarks of Qwen3-8B. The highest values after RL are **bolded**.

Method	τ-bench Retail	τ-bench Airline	ID Avg.	GPQA	ARC	MMLU	OOD Avg.	Avg.
ModSFT+RL	**78.3**	16.1	47.2	40.5	77.9	**62.5**	60.3	53.8
OverSFT+RL	70.3	**19.9**	45.1	**42.0**	75.9	61.7	59.9	52.5
DFT+RL	75.3	12.2	43.8	38.8	76.8	**62.5**	59.3	51.6
Rejuvenation+RL	77.0	17.5	**47.3**	41.0	**78.2**	62.4	**60.5**	**53.9**

gap. (3) Rejuvenation brings the ID average back to 19.7 and the OOD average to 15.3. The overall average recovers from 14.9 to 17.5, compared with 17.0 for ModSFT+RL, showing that an over-trained checkpoint can again serve as a useful initialization for RL.

4.2.1 Does Recovery Transfer to Agentic RL?

Table 3 summarizes the agentic results on Qwen3-8B. Rejuvenation lifts the Retail success rate from 70.3 to 77.0, close to ModSFT+RL (78.3). On Airline, it reaches 17.5, between ModSFT+RL (16.1) and OverSFT+RL (19.9). The overall average recovers from 52.5 to 53.9, compared with 53.8 for ModSFT+RL, supporting recovery in agentic RL despite differences across individual tasks.

4.3 Rejuvenation Widens the Viable Handoff Window

We further study whether Rejuvenation is specific to severely over-trained checkpoints or can also be applied to checkpoints with different SFT degrees. As shown in Figure 6, Rejuvenation recovers SFT models at different training stages on both EvoLM-4B and Qwen3-8B. Extending SFT from 2 to 32 epochs lowers the final average of direct RL from 17.0 to 14.9 on EvoLM-4B and from 53.8 to 52.5 on Qwen3-8B. With Rejuvenation, the averages remain within 16.7–17.6 and 53.7–54.3, respectively, across all five checkpoints, near their respective ModSFT+RL references (17.0 and 53.8). The performance range across checkpoints shrinks from 2.1 to 0.9 points on EvoLM-4B and from 1.3 to 0.6 points on Qwen3-8B. These results show that recovery extends beyond a single severely over-trained checkpoint: Rejuvenation widens the viable handoff window, allowing checkpoints from different SFT stages to serve as useful initializations for RL and making the SFT-to-RL handoff less sensitive to training duration.

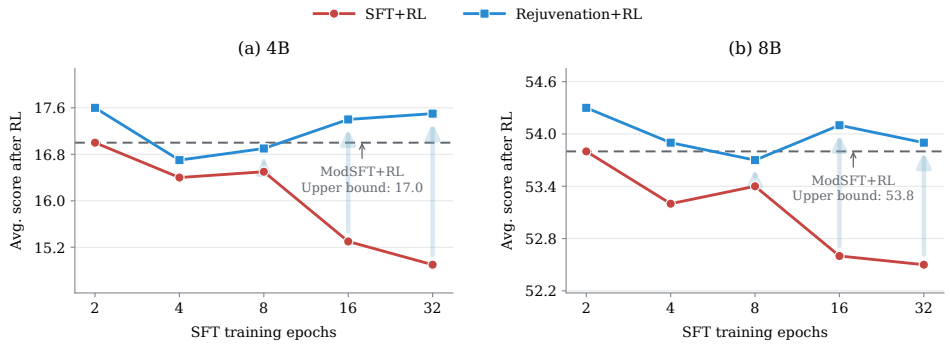


Figure 6: Final RL performance from checkpoints at 2, 4, 8, 16, and 32 SFT epochs on (a) EvoLM-4B and (b) Qwen3-8B, with and without Rejuvenation. We report the overall average (Avg.) using the same metric as Tables 2 and 3. The dashed lines mark the 2-epoch ModSFT+RL references.

4.4 Where Does the Recovery Come From?

Components. To understand the contribution of each component in Rejuvenation, we ablate it on EvoLM-4B. As shown in Table 4, base-anchored fusion alone brings most of the gain, recovering the overall average from 14.9 to 17.0. Adding attribution-guided neuron reset further improves it to 17.5, especially on OOD benchmarks, consistent with the complementary roles discussed in Section 3.4.

Recovery of output diversity. Figure 10 visualizes how token entropy changes before and after neuron reset. Over-confident predictions leave few alternative continuations for RL to explore. Resetting high-attribution neurons relaxes these predictions and restores diversity at the affected positions,

Table 4: Evaluation results on ID and OOD benchmarks after RL training. The highest values are **bolded**.

Method	GSM8K	MATH	AMC23	Miner.	Olymp.	ID Avg.	GPQA	ARC	MMLU	OOD Avg.	Avg.
OverSFT	45.2	22.0	6.6	9.9	4.1	17.5	9.9	21.3	5.6	12.2	14.9
OverSFT w/Fusion	46.9	**26.6**	6.3	**12.2**	**4.3**	19.3	**14.7**	22.7	**7.0**	14.8	17.0
Rejuvenation	**49.1**	26.2	**8.0**	10.8	4.2	**19.7**	12.9	**26.2**	6.9	**15.3**	**17.5**

giving reward-driven updates more room to reshape the policy. Together with the performance recovery in Table 4, these results connect the recovery of output diversity to the model's ability to learn again during RL.

Fusion Weight. Table 5 compares different fusion weights on EvoLM-4B. A small α (e.g., 0.4) yields lower ID performance but stronger OOD performance, whereas a large α (e.g., 0.6) improves ID performance at the cost of OOD generalization. We use $\alpha = 0.5$ in the main comparison to balance the two effects.

Table 5: Evaluation results on ID and OOD benchmarks with different fusion weight α. The highest values are **bolded**.

Value	GSM8K	MATH	AMC23	Miner.	Olymp.	ID Avg.	GPQA	ARC	MMLU	OOD Avg.	Avg.
0.4	47.7	23.9	6.2	10.3	4.5	18.5	12.8	**26.4**	5.9	**15.0**	16.8
0.5	46.9	26.6	6.3	**12.2**	4.3	19.3	**14.7**	22.7	**7.0**	14.8	**17.0**
0.6	**50.1**	**26.8**	**6.8**	9.8	**5.0**	**19.7**	8.8	21.0	5.9	11.9	15.8

Reset Percentage. Table 6 compares different reset percentages ρ. A small ρ of 0.5% gives the highest ID average, while 1% and 2% improve OOD generalization. We use $\rho = 1\%$ in the main comparison to balance recovery and retention.

Table 6: Evaluation results on ID and OOD benchmarks with different reset percentage ρ. The highest values are **bolded**.

Value	GSM8K	MATH	AMC23	Miner.	Olymp.	ID Avg.	GPQA	ARC	MMLU	OOD Avg.	Avg.
0.5%	**49.1**	25.9	**10.7**	12.0	3.9	**20.3**	11.6	21.3	6.0	13.0	16.7
1%	**49.1**	**26.2**	8.0	10.8	4.2	19.7	12.9	**26.2**	6.9	**15.3**	**17.5**
2%	48.9	25.5	7.3	**12.2**	**5.4**	19.9	**13.7**	23.4	**7.3**	14.8	17.3

5 Conclusion

In this paper, we identify a failure mode where excessive SFT training leads to loss of plasticity, which limits the effectiveness of subsequent RL optimization. Through analysis of parameter changes, token distributions, and RL training dynamics, we show that over-trained models become over-confident and harder to update. To address this issue, we introduce a two-stage method that includes global model fusion and neuron reset. These components help smooth the parameter space and restore diversity in key parts of the model, making it more amenable to further optimization during RL. Empirical results on mathematical and agentic tasks demonstrate the effectiveness of our method. Our approach recovers RL performance across different SFT stages, allowing checkpoints from a wider range of training durations to serve as useful initializations for RL. Our findings highlight the importance of model plasticity in the SFT-then-RL pipeline. We hope this work can motivate future research on understanding optimization dynamics in post-training and developing more robust training strategies.

Limitations. Although our method effectively restores the plasticity of the over-trained SFT models, both model fusion and neuron reset require the access to the base model for reference. We will explore how to restore the plasticity of over-trained models without the base model in the future work.

AI USE STATEMENT

In this work, we used generative AI tools for assisting with translation. We have not used generative AI tools for generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods, cleaning and reformatting datasets, supporting qualitative and thematic data analysis, or interpreting results. Additionally, we used generative AI tools for editing a research paper to improve readability. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have provided the experimental details in Section 4 and Appendix B. And we will release our code after paper acceptance.

REFERENCES

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Jack Chen, Fazhong Liu, Naruto Liu, Yuhan Luo, Erqu Qin, Harry Zheng, Tian Dong, Haojin Zhu, Yan Meng, and Xiao Wang. Step-wise adaptive integration of supervised fine-tuning and reinforcement learning for task-specific llms. *arXiv preprint arXiv:2505.13026*, 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. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 10818–10838. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/chu25c.html.

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. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8493–8502, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.581. URL https://aclanthology.org/2022.acl-long.581/.

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(8026): 768–774, 2024.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Yuqian Fu, Tinghong Chen, Jiajun Chai, Xihuai Wang, Songjun Tu, Guojun Yin, Wei Lin, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. SRFT: A single-stage method with supervised and

reinforcement fine-tuning for reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=n6E0r6kQWQ.

Wangjie Gan, Miao Pan, Linbo Xi, Wenqi Zhang, Jintao Chen, Jianwei Yin, and Xuhong Zhang. Gft: From imitation to reward fine-tuning with unbiased group advantages and dynamic coefficient rectification. *arXiv preprint arXiv:2604.14258*, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Tessa Han, Sebastian Bordt, Hanlin Zhang, and Sham Kakade. Weight decay improves language model plasticity. *arXiv preprint arXiv:2602.11137*, 2026.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.211. URL https://aclanthology.org/2024.acl-long.211/.

Zeyu Huang, Tianhao Cheng, Zihan Qiu, Zili Wang, Yinghui Xu, Edoardo M Ponti, and Ivan Titov. Blending supervised and reinforcement fine-tuning with prefix sampling. *arXiv preprint arXiv:2507.01679*, 2025.

Zixian Huang, Kaichen Yang, Xu Huang, Feiyang Hao, Qiming Ge, Bowen Li, He Du, Kai Chen, and Qipeng Guo. How to fine-tune a reasoning model? a teacher-student cooperation framework to synthesize student-consistent sft data. *arXiv preprint arXiv:2604.14164*, 2026.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=6t0Kwf8-jrj.

Jett Janiak, Can Rager, James Dao, and Yeu-Tong Lau. An adversarial example for direct logit attribution: Memory management in GELU-4L. In Yonatan Belinkov, Najoung Kim, Jaap Jumelet, Hosein Mohebbi, Aaron Mueller, and Hanjie Chen (eds.), *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 232–237, Miami, Florida, US, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.blackboxnlp-1.15. URL https://aclanthology.org/2024.blackboxnlp-1.15/.

Hangzhan Jin, Sitao Luan, Sicheng Lyu, Guillaume Rabusseau, Reihaneh Rabbany, Doina Precup, and Mohammad Hamdaqa. Rl fine-tuning heals ood forgetting in sft. *arXiv preprint arXiv:2509.12235*, 2025a.

Hangzhan Jin, Sicheng Lv, Sifan Wu, and Mohammad Hamdaqa. Rl is neither a panacea nor a mirage: Understanding supervised vs. reinforcement learning fine-tuning for llms. *arXiv preprint arXiv:2508.16546*, 2025b.

Feiyang Kang, Michael Kuchnik, Karthik Padthe, Marin Vlastelica, Ruoxi Jia, Carole-Jean Wu, and Newsha Ardalani. Quagmires in SFT-RL post-training: When high SFT scores mislead and what to use instead. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=uLM3BfKo19.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph 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*, SOSP '23, pp. 611–626, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

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 S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 3843–3857. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/18abbeef8cfe9203fdf9053c9c4fe191-Paper-Conference.pdf.

Xinran Li, Guangda Huzhang, Siqi Shen, Qing-Guo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, and Jun Zhang. Getting your LLMs ready for reinforcement learning with lightweight SFT. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=yezWGJmODg.

Ziniu Li, Congliang Chen, Tian Xu, Zeyu Qin, Jiancong Xiao, Zhi-Quan Luo, and Ruoyu Sun. Preserving diversity in supervised fine-tuning of large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=NQEe7B7bSw.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Alexis Limozin, Eduard Durech, Torsten Hoefler, Imanol Schlag, and Valentina Pyatkin. Sft-then-rl outperforms mixed-policy methods for llm reasoning. *arXiv preprint arXiv:2604.23747*, 2026.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. ProRL: Prolonged reinforcement learning expands reasoning boundaries in large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=YPsJha5HXQ.

Mingyang Liu, Gabriele Farina, and Asuman E. Ozdaglar. UFT: Unifying supervised and reinforcement fine-tuning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=usOkGv1S7M.

Tao Liu, Taiqiang Wu, Runming Yang, Shaoning Sun, Junjie Wang, and Yujiu Yang. Profit: Leveraging high-value signals in sft via probability-guided token selection. *arXiv preprint arXiv:2601.09195*, 2026.

Yihao Liu, Shuocheng Li, Lang Cao, Yuhang Xie, Mengyu Zhou, Haoyu Dong, Xiaojun Ma, Shi Han, and Dongmei Zhang. Superrl: Reinforcement learning with supervision to boost language model reasoning. *arXiv preprint arXiv:2506.01096*, 2025c.

Xingtai Lv, Yuxin Zuo, Youbang Sun, Hongyi Liu, Yuntian Wei, Zhekai Chen, Xuekai Zhu, Kaiyan Zhang, Bingning Wang, Ning Ding, et al. Towards a unified view of large language model post-training. *arXiv preprint arXiv:2509.04419*, 2025.

Lu Ma, Hao Liang, Meiyi Qiang, Lexiang Tang, Xiaochen Ma, Zhen Hao Wong, Junbo Niu, Chengyu Shen, Runming He, Yanhao Li, Wentao Zhang, and Bin CUI. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=LzCBLrNoyM.

MAA. American mathematics contest 12 (amc 12), November 2023. URL https://artofproblemsolving.com/wiki/index.php/AMC_12_Problems_and_Solutions. Accessed: 2026-04-30.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/2c601ad9d2ff9bc8b282670cdd54f69f-Paper.pdf.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJGCiw5gl.

OpenAI. Learning to reason with llms, 2024. URL https://openai.com/index/learning-to-reason-with-llms. Accessed: 2026-04-30.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Zhenting Qi, Fan Nie, Alexandre Alahi, James Zou, Himabindu Lakkaraju, Yilun Du, Eric P. Xing, Sham M. Kakade, and Hanlin Zhang. EvoLM: In search of lost language model training dynamics. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=B6bE2GC71a.

Chongli Qin and Jost Tobias Springenberg. Supervised fine tuning on curated data is reinforcement learning (and can be improved). *arXiv preprint arXiv:2507.12856*, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, pp. 1279–1297, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400711961. doi: 10.1145/3689031.3696075. URL https://doi.org/10.1145/3689031.3696075.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Hao Wang, Hao Gu, Hongming Piao, Kaixiong Gong, Yuxiao Ye, Xiangyu Yue, Sirui Han, Yike Guo, and Dapeng Wu. Learning while staying curious: Entropy-preserving supervised fine-tuning via adaptive self-distillation for large reasoning models. *arXiv preprint arXiv:2602.02244*, 2026.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=yfcpdY4gMP.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi

Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 95266–95290. Curran Associates, Inc., 2024. doi: 10.52202/079017-3018. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/ad236edc564f3e3156e1b2feafb99a24-Paper-Datasets_and_Benchmarks_Track.pdf.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya 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 Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 23965–23998. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/wortsman22a.html.

Yongliang Wu, Yizhou Zhou, Zhou Ziheng, Yingzhe Peng, Xinyu Ye, Xinting Hu, Wenbo Zhu, Lu Qi, Ming-Hsuan Yang, and Xu Yang. On the generalization of SFT: A reinforcement learning perspective with reward rectification. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Lv7PjbcaMi.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=vO8LLoNWWk.

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, 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. *arXiv preprint arXiv:2412.15115*, 2024.

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.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik R Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=roNSXZpUDN.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, YuYue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=2a36EMSSTp.

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 *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=4OsgYD7em5.

Dylan Zhang, Yufeng Xu, Haojin Wang, Qingzhi Chen, and Hao Peng. Good sft optimizes for sft, better sft prepares for reinforcement learning. *arXiv preprint arXiv:2602.01058*, 2026a.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, Yu Fu, Xingtai Lv, Yuchen Zhang, Sihang Zeng, Shang Qu, Haozhan Li, Shijie Wang, Yuru Wang, Xinwei Long, Fangfu Liu, Xiang Xu, Jiaze Ma, Xuekai Zhu, Ermo Hua, Yihao Liu, Zonglin Li, Huayu Chen, Xiaoye Qu, Yafu Li, Weize Chen, Zhenzhao Yuan, Junqi Gao, Dong Li, Zhiyuan Ma, Ganqu Cui, Zhiyuan Liu, Biqing Qi, Ning Ding, and Bowen Zhou. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*, 2025a.

Wenhao Zhang, Yuexiang Xie, Yuchang Sun, Yanxi Chen, Guoyin Wang, Yaliang Li, Bolin Ding, and Jingren Zhou. On-policy RL meets off-policy experts: Harmonizing supervised fine-tuning and reinforcement learning via dynamic weighting. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=dCm9bBrk5d.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 10495–10516, Vienna, Austria, July 2025b. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.547. URL https://aclanthology.org/2025.findings-acl.547/.

Jian Zhao, Runze Liu, Kaiyan Zhang, Zhimu Zhou, Junqi Gao, Dong Li, Jiafei Lyu, Zhouyi Qian, Biqing Qi, Xiu Li, and Bowen Zhou. Genprm: Scaling test-time compute of process reward models via generative reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(41): 34932–34940, Mar. 2026. doi: 10.1609/aaai.v40i41.40797. URL https://ojs.aaai.org/index.php/AAAI/article/view/40797.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. Sglang: Efficient execution of structured language model programs. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 62557–62583. Curran Associates, Inc., 2024a. doi: 10.52202/079017-2000. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/724be4472168f31ba1c9ac630f15dec8-Paper-Conference.pdf.

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

He Zhu, Junyou Su, Peng Lai, Ren Ma, Wenjia Zhang, Linyi Yang, and Guanhua Chen. Anchored supervised fine-tuning. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=PORko7QT64.

Wenhong Zhu, Ruobing Xie, Rui Wang, Xingwu Sun, Di Wang, and Pengfei Liu. Proximal supervised fine-tuning. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=hQtwQqYikp.

Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. slime: An llm post-training framework for rl scaling. https://github.com/THUDM/slime, 2025. GitHub repository. Corresponding author: Xin Lv. Accessed: 2026-04-30.

A Full Algorithm

Algorithm 1 `Rejuvenation`: Plasticity Recovery for Over-Trained SFT Models

Input: Base model $\theta_{\texttt{Base}}$, over-trained model θ_{over}, fusion weight α, reset ratio ρ

1: Apply global model fusion via equation 1
2: Compute neuron attribution scores at selected over-confident tokens
3: For each selected target token, select the top ρ fraction of neurons globally by positive attribution score
4: Let $\mathcal{S}$ be the union of selected neurons over all calibration tokens
5: Reset selected neurons in θ_{fuse} to their base values

Output: recovered model θ_{rec}

B Experimental Details

Models and Baselines. We adopt EvoLM-4B (Qi et al., 2025) as the base model for mathematical tasks since it is pre-trained on a controlled corpus without evaluation data contamination, which makes the gain from RL more reliable. For agentic tasks, we use Qwen3-8B (Yang et al., 2025), a strong general-purpose backbone widely used in tool-use scenarios. To isolate the effect of plasticity recovery on the SFT-to-RL handoff, we compare `Rejuvenation` against the following baselines: (1) `ModSFT`: a moderately trained SFT checkpoint (epoch=2), which serves as a strong reference for the SFT-to-RL handoff; (2) `OverSFT`: the over-trained SFT checkpoint (epoch=32) that exhibits plasticity loss and is the target for our recovery method; (3) `DFT` (Wu et al., 2026): a representative regularized SFT objective that re-weights the cross-entropy loss with token probabilities to mitigate over-confidence during SFT. Following the same protocol, we report the corresponding +RL results obtained by running the same RL recipe on top of each SFT variant.

Mathematical SFT Training Details. For EvoLM-4B, we conduct SFT with LlamaFactory (Zheng et al., 2024b) on a 100k subset sampled from a 500k mathematical SFT corpus. The base checkpoint is the EvoLM-4B pre-trained model trained with controlled mixing of FineWeb and FineMath data (without contamination from the evaluation benchmarks). All variants share the same data, the same context window, and the same data ordering, so any difference between `ModSFT` and `OverSFT` is purely a function of training duration. We use AdamW (Kingma, 2014) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, a constant learning rate of 3.0×10^{-6}, and *no* learning-rate warmup, decay, or weight decay. We deliberately disable schedules and weight decay so that excessive SFT can manifest its full effect on parameter drift, weight magnitude, and token entropy. `ModSFT` is the checkpoint after 2 epochs and `OverSFT` is the checkpoint after 32 epochs over the same SFT subset. For the `DFT` baseline, we keep the same data, the same total epochs (32), and the same optimizer settings as `OverSFT`, and only replace the cross-entropy loss with the probability-reweighted DFT objective (Wu et al., 2026). This isolates the effect of the loss formulation from the effect of the SFT trajectory length.

Agentic SFT Training Details. For Qwen3-8B, the SFT stage is performed with slime (Zhu et al., 2025) on the τ-bench Retail training split, where supervision trajectories are generated from a stronger Qwen3-30B-A3B teacher and saved as multi-turn assistant-tool conversations. We use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$, a constant learning rate of 3.0×10^{-6}, weight decay 0.1, a global batch size of 16, and a maximum response length of 4,096 tokens. We train for 32 epochs with the Qwen3 chat template (`enable_thinking`=False) and the Qwen3 tool-aware loss mask, so that loss is only computed on assistant turns. The corresponding `ModSFT` and `OverSFT` checkpoints follow the same epoch-based convention as in the math setting. The `DFT` baseline is constructed in the same way as in the math setting, by replacing the cross-entropy loss with the DFT objective while keeping all other hyper-parameters identical.

Mathematical RL Training Details. We use verl (Sheng et al., 2025) with GRPO (Shao et al., 2024) as the RL backbone. The actor is initialized from the corresponding SFT or rejuvenated checkpoint, and the reference policy is kept frozen during RL. The training prompt batch size is 512 with a mini-batch size of 128, and we sample 8 responses per prompt with temperature 1.0 and top-p 1.0. The maximum prompt length is 512 and the maximum response length is 1,024 tokens. For optimization, we use AdamW with a constant learning rate of 1×10^{-6} and a global gradient clip of 1.0. We keep the GRPO clipping range symmetric ($\varepsilon_{\text{low}} = \varepsilon_{\text{high}} = 0.2$), use KL loss with coefficient 0.001, and adopt the `seq-mean-token-mean` loss aggregation. Rewards are computed by the

Math-Verify[3] verifier on the boxed final answer, with the reward function returning 1 for correct answers and -1 otherwise. Validation is run every 50 steps with temperature 0.6 and top-p 1.0. Each run uses 1 node of 8×NVIDIA H800 GPUs, FSDP2 sharding, and dynamic batching. Rollouts are served by vLLM (Kwon et al., 2023).

Agentic RL Training Details. For agentic RL, we use slime (Zhu et al., 2025) with Megatron-LM (Shoeybi et al., 2019) backend for training and SGLang (Zheng et al., 2024a) for inference. For each iteration we sample 12 prompts from the τ-bench Retail training set with 8 trajectories per prompt and a global batch size of 96, with maximum response length 1,024 tokens and temperature 1.0. To produce realistic multi-turn behavior, we deploy a separate Qwen3-4B-Instruct-2507 (Yang et al., 2025) as the user simulator, served locally via SGLang, and call it through an OpenAI-compatible API. We use AdamW with learning rate 1×10^{-6}, constant LR schedule, and CPU optimizer offloading. We use GRPO algorithm and set the KL loss coefficient and the entropy coefficient both to 0, and use clipping ranges with $\varepsilon_{\text{low}} = 0.2$ and $\varepsilon_{\text{high}} = 0.28$. Tensor-model-parallel size is 2, sequence parallelism is enabled, and we use full activation recomputation with dynamic batching (up to 2,048 tokens per GPU). We train for up to 200 rollouts, save and evaluate every 20 rollouts, and balance trajectories across data-parallel ranks with the standard non-zero-reward-std dynamic filter. Each run uses 6 GPUs for the policy and the user simulator combined, with the user simulator pinned to a separate set of GPUs.

For both math and agentic RL, we compare SFT and rejuvenated initializations under the same RL recipe. Baselines start from `ModSFT`, `OverSFT`, or the `DFT` checkpoint, whereas `Rejuvenation` starts from the rejuvenated checkpoint produced by Algorithm 1. All RL hyper-parameters, data, prompts, and seeds are kept identical across initializations to ensure a fair comparison.

Evaluation Details. For mathematical reasoning, we evaluate on five widely used in-distribution (ID) benchmarks: GSM8K (Cobbe et al., 2021), MATH-500 (Lightman et al., 2024), AMC23 (MAA, 2023), Minerva Math (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024). For agentic tasks, we evaluate on τ-bench (Yao et al., 2025) Retail and Airline. We further evaluate on three out-of-distribution (OOD) benchmarks: GPQA-Diamond (Rein et al., 2024), ARC-Challenge (Clark et al., 2018), and MMLU-Pro (Wang et al., 2024) following Yan et al. (2025). We report Pass@1 averaged over multiple samples for mathematical and general benchmarks, and task success rates for agentic tasks. All scores are percentages. ID Avg. and OOD Avg. are unweighted averages within their respective benchmark groups; the overall Avg. is their equally weighted mean.

For ID and OOD evaluation we use the same inference backends as in training: vLLM for EvoLM-4B and SGLang for Qwen3-8B. We use a sampling temperature of 0.6 and top-p of 1.0 for all benchmarks. For benchmarks with limited problem counts, we average Pass@1 over multiple samples per problem (K=32 for AMC23, K=4 for MATH-500, Minerva, OlympiadBench, GPQA, ARC, K=2 for GSM8K). Mathematical answers are graded with the same Math-Verify verifier used during RL training, and multiple-choice OOD benchmarks (GPQA, ARC, MMLU-Pro) are graded by the official answer-matching scripts shipped with each dataset.

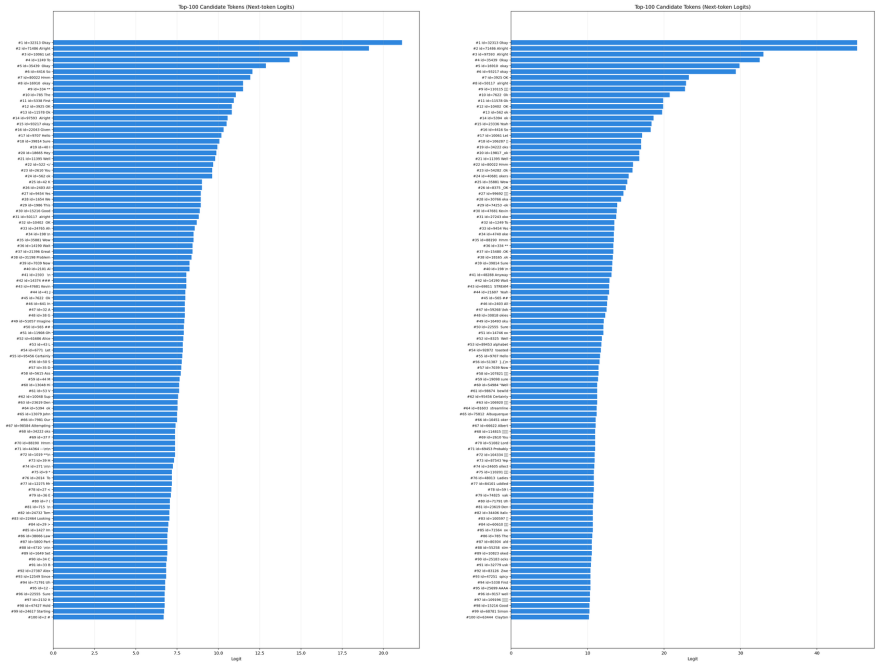
C Additional Results

Logit Distribution. Figure 7 contrasts the next-token logits of `ModSFT` and `OverSFT` for the same input. The highest-scoring candidates are more sharply separated from the remaining tokens in `OverSFT`, consistent with more concentrated next-token predictions.

Logit Attribution. The attribution scores are highly uneven across both modules and neurons. As shown in Figure 8, a small number of modules have much larger positive contributions to the target logits. The top-ranked neurons also contribute substantially more than other neurons shown in the ranking. These patterns motivate targeting high-attribution neurons in the reset step.

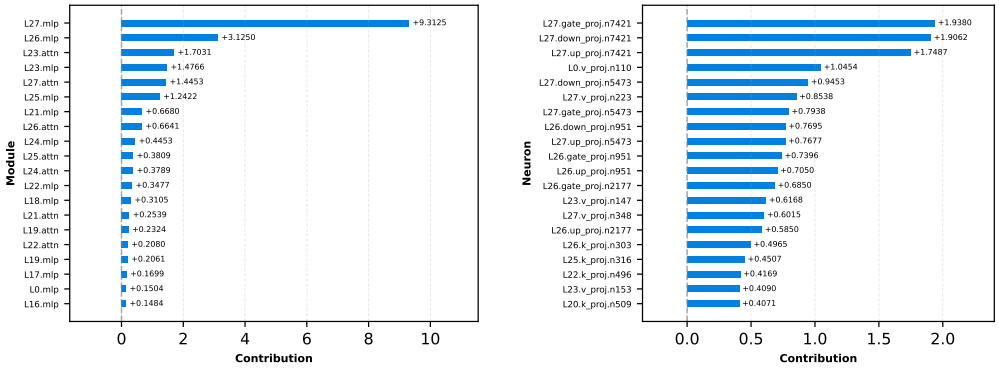
Selected Tokens in Logit Attribution. We visualize the selected tokens in Figure 9. They include logical connectives and words that organize reasoning, such as *Since*, *First*, and *conclude*, alongside mathematical and formatting tokens. This is consistent with prior work linking reasoning-related tokens to RL improvement (Wang et al., 2025). Restoring diversity at these positions may help reopen alternative reasoning continuations during RL.

[3] https://github.com/huggingface/Math-Verify



(a) ModSFT (b) OverSFT

Figure 7: Token logit comparison between normal SFT models and over-trained models: (a) ModSFT; (b) OverSFT.



(a) module level (b) neuron level

Figure 8: Logit attribution ranking: (a) module level; (b) neuron level.

How does the entropy change? To better understand how neuron reset restores output diversity, we visualize token entropy before and after reset in Figure 10. Resetting high-attribution neurons relaxes over-confident predictions and reopens alternative token choices for exploration during RL. This recovery of output diversity provides an output-space explanation for the performance recovery in Section 4.

Selected neurons/modules. Figure 11 visualizes the distribution of reset neurons over the whole model. Reset ratios vary across layers and modules, with higher ratios in `k_proj` and `v_proj`, especially in several later layers.

Computational Cost. `Rejuvenation` is a one-shot, post-hoc operation and introduces negligible overhead compared with the SFT or RL stages. The fusion step is a single element-wise interpolation

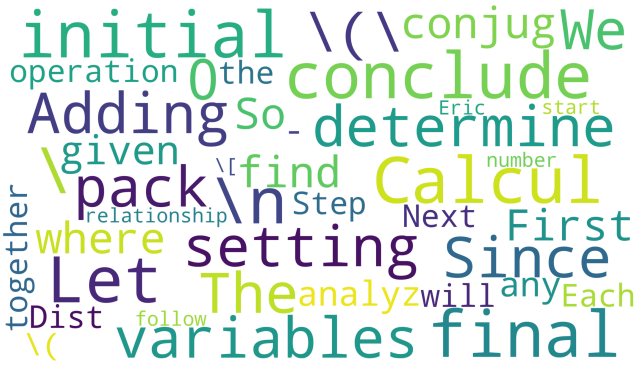
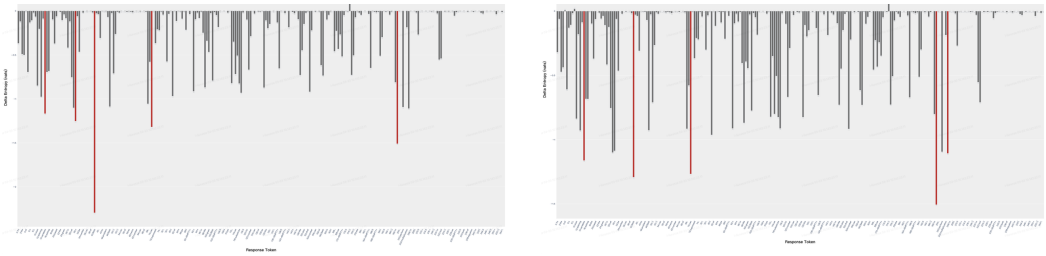


Figure 9: Wordcloud of selected tokens in logit attribution.



(a) Before neuron reset

(b) After neuron reset

Figure 10: Comparison of token entropy: (a) Before neuron reset; (b) After neuron reset.

between two checkpoints. The neuron reset step requires one forward pass over a small calibration set to compute DLA scores and a single masked copy from the base weights into the fused checkpoint. Both steps are run on a single NVIDIA H800 GPU and finish in 3 minutes for EvoLM-4B and within 5 minutes for Qwen3-8B, which is orders of magnitude smaller than the cost of either re-running SFT from a different stopping point or relaunching RL from multiple SFT checkpoints.

Parameter Changes. To complement the per-module visualization in Section 3.1, we provide additional layer-wise visualizations of the parameter shift induced by SFT and the subsequent RL stage. Figure 12 and Figures 13-14 cover representative attention projections across early and late layers. The patterns are consistent with those reported in the main text: `OverSFT` introduces sparse but extreme spikes that are concentrated in a small subset of parameters, while `ModSFT` produces relatively smooth and small-magnitude updates. Subsequent RL on `OverSFT` does not fully remove these spikes under our RL recipe, consistent with the limited recovery from over-trained initializations.

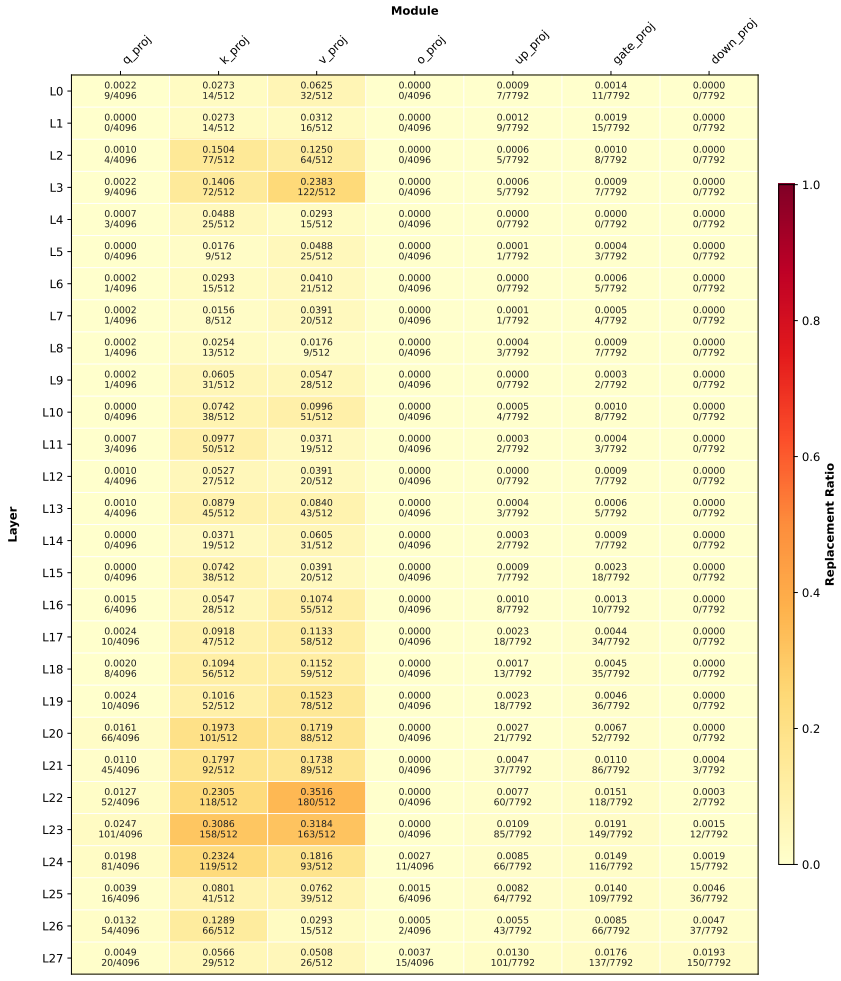


Figure 11: Per layer and per module visualization of neuron reset ratio in EvoLM-4B.

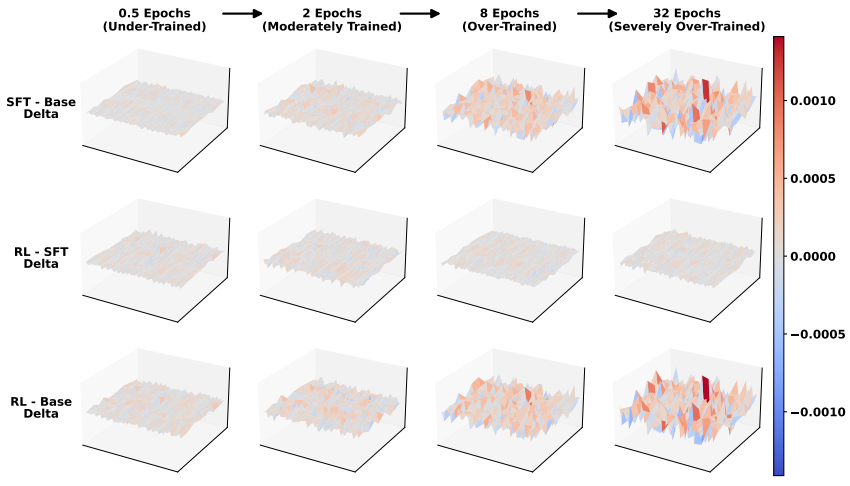


Figure 12: Parameter changes of `layers.0.self_attn.k_proj.weight` induced by SFT and RL.

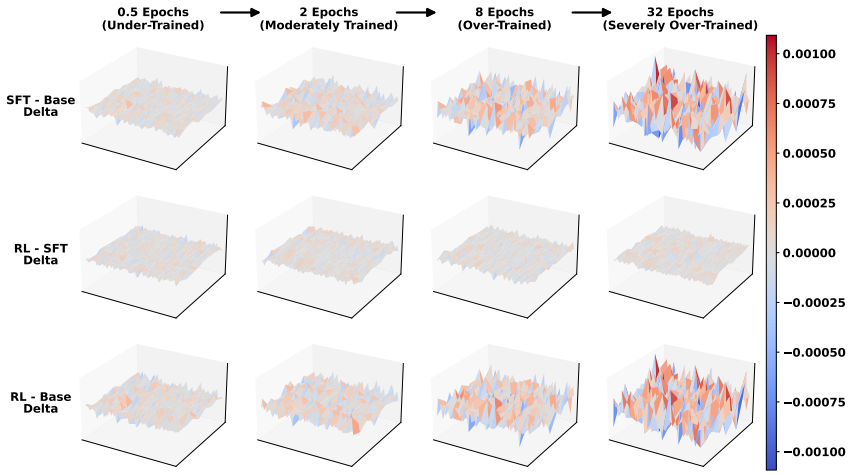


Figure 13: Parameter changes of `layers.27.self_attn.v_proj.weight` induced by SFT and RL.

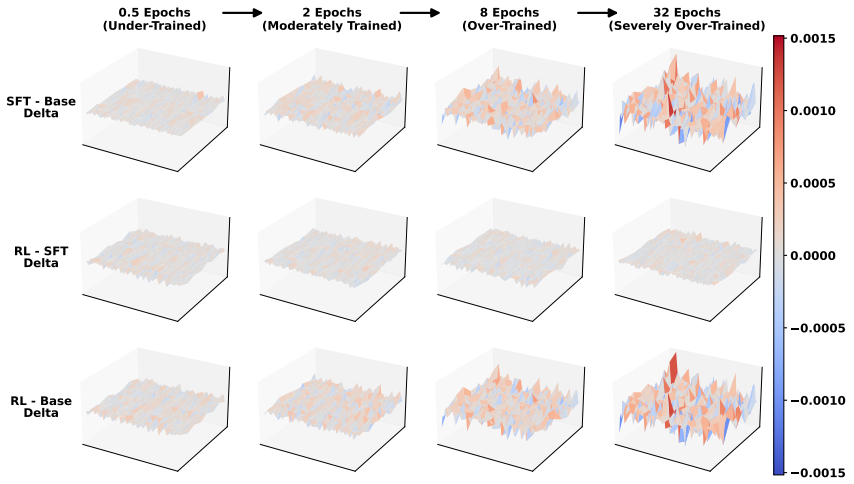


Figure 14: Parameter changes of `layers.27.self_attn.k_proj.weight` induced by SFT and RL.