RETAINING BY DOING: THE ROLE OF ON-POLICY DATA IN MITIGATING FORGETTING

Howard Chen **Noam Razin** **Karthik Narasimhan** **Danqi Chen**
Princeton Language and Intelligence, Princeton University
{howardchen,noamrazin,karthikn,danqic}@cs.princeton.edu

ABSTRACT

Adapting language models (LMs) to new tasks via post-training carries the risk of degrading existing capabilities—a phenomenon classically known as *catastrophic forgetting*. In this paper, toward identifying guidelines for mitigating this phenomenon, we systematically compare the forgetting patterns of two widely adopted post-training methods: supervised fine-tuning (SFT) and reinforcement learning (RL). Our experiments reveal a consistent trend across LM families (Llama, Qwen) and tasks (instruction following, general knowledge, and arithmetic reasoning): RL leads to less forgetting than SFT while achieving comparable or higher target task performance. To investigate the cause for this difference, we consider a simplified setting in which the LM is modeled as a mixture of two distributions, one corresponding to prior knowledge and the other to the target task. We identify that the *mode-seeking* nature of RL, which stems from its use of *on-policy* data, enables keeping prior knowledge intact when learning the target task. We then verify this insight by demonstrating that the use on-policy data underlies the robustness of RL to forgetting in practical settings, as opposed to other algorithmic choices such as the KL regularization or advantage estimation. Lastly, as a practical implication, our results highlight the potential of mitigating forgetting using *approximately* on-policy data, which can be substantially more efficient to obtain than fully on-policy data.[1]

1 INTRODUCTION

Adapting language models (LMs) to new target tasks during post-training carries the risk of eroding previously acquired capabilities—a phenomenon known as *catastrophic forgetting* (McCloskey & Cohen, 1989; Kirkpatrick et al., 2017). Such forgetting has been reported to occur when training LMs to follow instructions via supervised fine-tuning (SFT) (Luo et al., 2023; Shi et al., 2024; Wu et al., 2024) or aligning them with human preferences via reinforcement learning (RL) (Bai et al., 2022; Ouyang et al., 2022). However, the understanding of how SFT and RL compare in terms of their susceptibility to forgetting remains limited.

In this work, we systematically compare the forgetting patterns of SFT and RL in order to identify principled guidelines for mitigating forgetting in LM post-training. We conduct a comprehensive study across instruction following, general knowledge, and arithmetic reasoning tasks, using Qwen 2.5 (Yang et al., 2024) and Llama 3 (Grattafiori et al., 2024) models of up to 8B scale. Our experiments reveal a consistent trend: SFT suffers from severe forgetting, whereas RL can achieve high target task performance without substantial forgetting (Figure 2).

We then investigate the cause for the relative robustness of RL to forgetting. At first glance, it may seem at odds with conventional wisdom. Namely, minimizing the cross-entropy loss via SFT is equivalent to minimizing the *forward KL* divergence with respect to the optimal policy, while maximizing the RL objective corresponds to minimizing the *reverse KL* (Korbak et al., 2022). Conventional wisdom presumes that the *mode-seeking* nature of reverse KL enables faster learning of target distributions (Chan et al., 2022; Tajwar et al., 2024b) at the cost of losing coverage of old modes, while the *mode-covering* forward KL should maintain probability mass across modes. We

[1]Code is available at: https://github.com/princeton-pli/retaining-by-doing

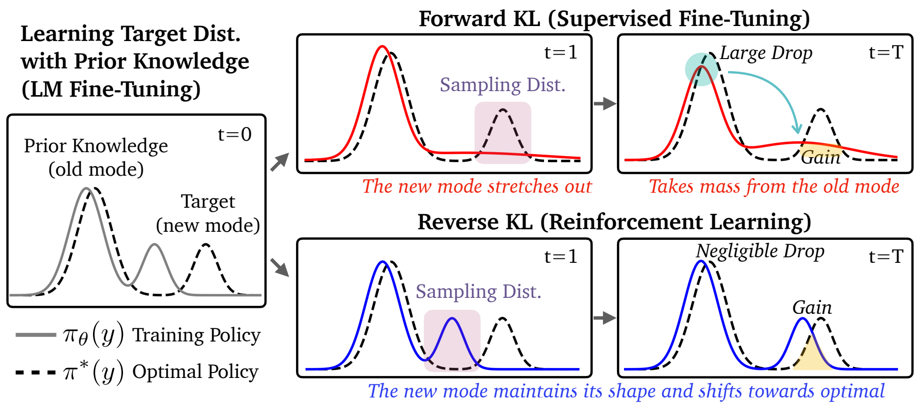
Learning Target Dist. with Prior Knowledge (LM Fine-Tuning)
t=0
Prior Knowledge (old mode)
Target (new mode)
$\pi_\theta(y)$ Training Policy
$\pi^*(y)$ Optimal Policy
Forward KL (Supervised Fine-Tuning)
t=1
Sampling Dist.
The new mode stretches out
Large Drop
t=T
Gain
Takes mass from the old mode
Reverse KL (Reinforcement Learning)
t=1
Sampling Dist.
Negligible Drop
t=T
Gain
The new mode maintains its shape and shifts towards optimal

Figure 1: Illustration of the forgetting dynamics for the forward KL objective, corresponding to SFT, and the reverse KL objective, corresponding to RL. Left: we model LM post-training as a mixture of two modes. The "old" mode represents prior knowledge and the "new" mode represents a target task. Initially, the old mode of the training policy π_θ roughly matches the old mode of the optimal policy π^*, but its additional "new" mode does not match the new target mode. The goal is for the training policy to match the optimal policy. Top right: minimizing forward KL first stretches the new mode of π_θ and then moves probability mass from the old mode to cover the target, leading to forgetting. Bottom right: in contrast, minimizing reverse KL maintains the shape of the old mode and covers the target distribution by shifting the new mode of π_θ.

reconcile this discrepancy by considering a simplified setting, where the target distribution is modeled as a mixture of two distributions: one representing the policy's prior knowledge and the other representing the target task. We show that, if the initial policy is uni-modal (*i.e.*, has a single mode), then SFT can in fact be more robust than RL to forgetting. However, if the initial policy is multi-modal (*i.e.*, has multiple modes), which is arguably the case for practical LMs, then mode-seeking RL leads to less forgetting than mode-covering SFT; see Figure 1 for an illustration.

The mode-seeking behavior of RL (*i.e.*, its accordance with reverse KL minimization) stems from the usage of *on-policy* data. Through extensive ablations, we empirically verify that this property underlies the robustness of RL to forgetting, as opposed to other algorithmic choices such as the advantage estimation or the application of KL regularization. Moreover, we explore what degree of on-policy data allows mitigating forgetting. We find that for SFT, while generating data only from the initial policy is not enough, *approximately on-policy* data generated at the start of each epoch can suffice for substantially reducing forgetting. This suggests a practical guideline for LM post-training: leveraging on-policy data, potentially sampled asynchronously or at the start of each epoch for improved efficiency, can reduce unintended disruption of the model's existing capabilities.

To summarize, our main contributions are:

- We demonstrate that RL is more robust to forgetting than SFT through extensive experiments on instruction following, general knowledge, and reasoning tasks, using LMs from different families and scales.
- We provide intuition for why the mode-seeking nature of RL, which stems from its use of on-policy data, can counterintuitively lead to less forgetting than mode-covering SFT.
- We corroborate this insight by demonstrating that the use of on-policy data underlies the robustness of RL to forgetting in practical settings, and highlight the potential of mitigating forgetting through approximately on-policy data, which can be substantially more efficient to obtain than fully on-policy data.

2 Forgetting in LM Post-Training

We begin by introducing notation and the metrics used to measure forgetting. Then, we empirically compare the forgetting patterns of supervised fine-tuning (SFT) and reinforcement learning (RL).

2.1 PRELIMINARIES

A language model (LM) is modeled by a policy $\pi_\theta(y \mid x)$, where the response y is generated conditioned on the prompt x. For a target task $\mathcal{T}$, we denote the optimal policy by $\pi^*(\cdot \mid x)$. In SFT, the cross-entropy loss is minimized with respect to ground truth responses y^* sampled from the optimal policy: $\mathcal{L}_{\text{SFT}}(\theta; x) := \sum_y -\pi^*(y \mid x) \log \pi_\theta(y \mid x)$. By contrast, in RL, the goal is to maximize the KL-regularized reward with respect to responses generated by the LM and a reward function $r(x, y) \in \{0, 1\}$[2] *i.e.*: $J_{\text{RL}}(\theta; x) := \mathbb{E}_{y \sim \pi_\theta(\cdot \mid x)}[r(x, y)] - \beta \cdot \text{KL}[\pi_\theta(\cdot \mid x) \,||\, \pi_{\theta_0}(\cdot \mid x)]$, where $\beta > 0$ and π_{θ_0} is the initial policy.

Forgetting and evaluation metrics. The initial policy π_{θ_0} is trained on a target task $\mathcal{T}$ for T optimization steps, resulting in the trained policy π_{θ_T}. This policy is evaluated using accuracy, which measures the fraction of correct outputs generated by π_{θ_T} for prompts associated with $\mathcal{T}$. We denote the accuracy of π_{θ_T} over $\mathcal{T}$ by $\mathcal{A}(\pi_{\theta_T}, \mathcal{T})$ and define the *target task gain* as $\Delta_g := \mathcal{A}(\pi_{\theta_T}, \mathcal{T}) - \mathcal{A}(\pi_{\theta_0}, \mathcal{T})$. We quantify forgetting, based on on a collection of tasks $\{\mathcal{T}'_j\}_{j=1}^M$, through the *non-target tasks drop* $\Delta_d := \frac{1}{M} \sum_{j=1}^M \mathcal{A}(\pi_{\theta_0}, \mathcal{T}'_j) - \mathcal{A}(\pi_{\theta_T}, \mathcal{T}'_j)$. During post-training, the aim is to achieve high target task gain while minimizing as much as possible the non-target tasks drop. For brevity, we will often refer to target task gain as *gain* and to non-target tasks drop as *drop*.

2.2 EXPERIMENTAL SETUP

Target tasks and evaluation. We consider three tasks, covering different capabilities: IFEval (Zhou et al., 2023; Lambert et al., 2024) for instruction following, MMLU (Hendrycks et al., 2021a) for general knowledge, and Countdown (Pan et al., 2025) for arithmetic reasoning. These target task datasets are split into training set and evaluation set as described in Appendix A.3. After training on one target task, we evaluate the model's performance on all the other tasks. We additionally include the following non-target tasks: MATH (Hendrycks et al., 2021b); two safety datasets, WildJailbreak (Jiang et al., 2024) and WildGuardTest (Han et al., 2024), since safety capabilities are often eroded through fine-tuning (Qi et al., 2024), making them highly suitable for measuring forgetting. In our RL experiments, correct generations are assigned a reward of 1 and incorrect generations are assigned a reward of 0.

Models and baselines. We use instruct models from the Llama 3 (Grattafiori et al., 2024) and Qwen 2.5 (Yang et al., 2024) families as the initial policies: Llama-3.2-1B-Instruct, Llama-3.1-8B-Instruct, Qwen-2.5-1.5B-Instruct, and Qwen-2.5-7B-Instruct. We compare two SFT variants and one RL method: **1) SFT**, which uses responses generated by Llama-3.3-70B-Instruct as ground truth responses; **2) Self-SFT**, which uses responses generated by the initial model (we keep only the correct responses based on the reward function (Zelikman et al., 2022)); and **3) RL**—we use GRPO (Shao et al., 2024), a common algorithm for tasks with verifiable outputs. For both SFT variants, the generated data was filtered using the reward function to include only examples with correct responses. We use Self-SFT as a baseline to represent the typical first step in the post-training pipeline when human labels are absent (Dong et al., 2023; Lambert et al., 2024). All models are trained for two epochs. Additional implementation details are provided in Appendix A.3.

2.3 RESULTS: SFT FORGETS MORE THAN RL

Figure 2 compares the target task gain and non-target tasks drop of the SFT variants and RL. We observe higher levels of forgetting in SFT compared to RL across datasets, model families, and sizes. In particular, we find:

- For Self-SFT, achieving a similar target accuracy gain to RL induces a significantly larger drop on non-target tasks.
- While SFT can achieve a higher performance gain than RL on the instruction following task, it induces an even larger drop on non-target tasks relative to Self-SFT.
- As shown in Figure 3, a high learning rate is typically required to reach high target performance for SFT, often at the cost of severe forgetting; a smaller learning rate reduces forgetting but fails to reach the same target performance even with more epochs.

[2] We use RL with verifiable reward (RLVR) throughout our experiments.

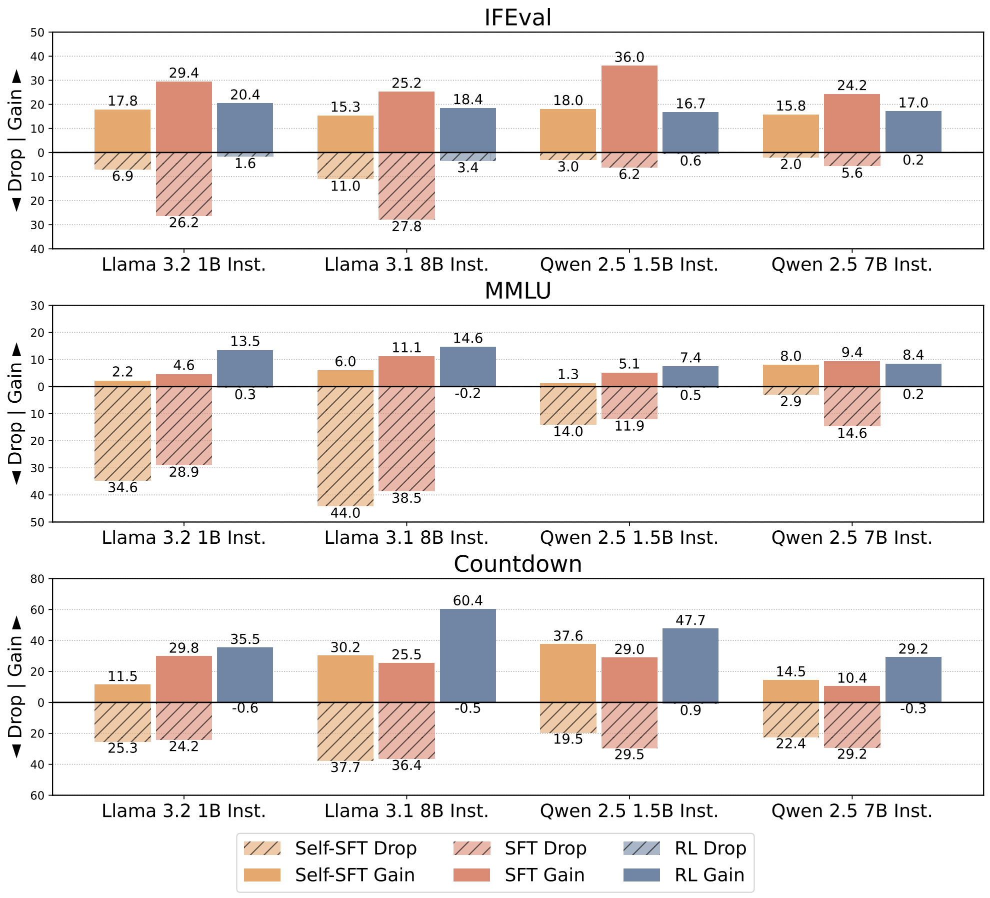

Figure 2: **SFT forgets more than RL across tasks and models.** We compare the **Gain** (solid bar) / **Drop** (shaded bar) across models and datasets for: *(1)* Self-SFT, which uses data generated from the initial policy; *(2)* SFT, which uses data generated by Llama-3.3-70B-Instruct; and *(3)* RL (GRPO). Gain (higher better) represents the accuracy increase on the target task, while drop (lower better) represents the average accuracy decrease on non-target tasks.

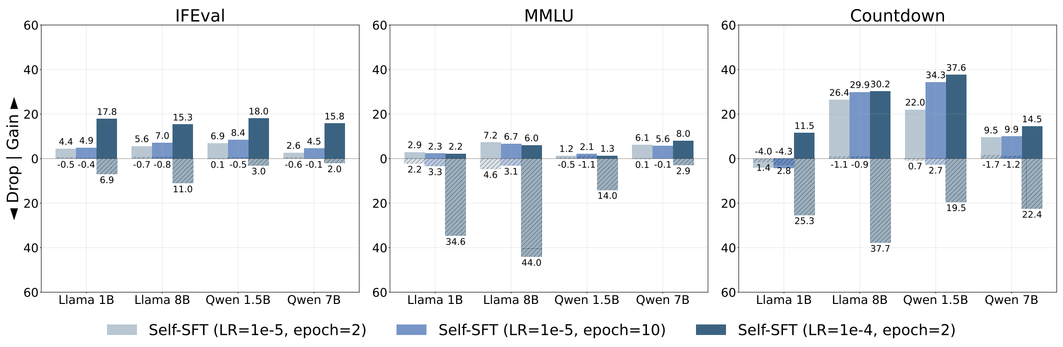

Figure 3: **SFT exhibits a tradeoff between target task performance and forgetting.** Comparison of Self-SFT runs with learning rates: $1e{-}5$ (small) and $1e{-}4$ (default), and training epochs $(2, 10)$.

Overall, both SFT variants exhibit a consistent tradeoff between target performance and forgetting, whereas RL improves target performance without noticeable drops on non-target tasks.

3 Understanding Forgetting Dynamics Through the Lens of KL

SFT and RL can be viewed as minimizing different directions of the KL divergence with respect to the optimal policy. Specifically, as reviewed in §3.1, SFT corresponds to forward KL minimization while RL corresponds to reverse KL minimization. Intuitively, a *mode-seeking* objective such as reverse KL should be more susceptible to forgetting: it moves probability mass quickly from one mode to another, whereas *mode-covering* forward KL should better maintain probability mass on all modes. This intuition is invalidated in light of the evidence presented in §2.3, showing that SFT causes more forgetting than RL. We address this discrepancy through an empirical analysis of a simplified setting with univariate Gaussian distributions. The analysis reveals that SFT can in fact lead to less forgetting than RL if the initial policy has a single mode (§3.2). However, in multi-modal scenarios that arguably mirror more closely LM fine-tuning, we show that the mode-seeking properties of RL result in higher degrees of robustness to forgetting (§3.3).

3.1 SFT and RL as KL Minimization

SFT as forward KL minimization (mode-covering). It is widely known that SFT is equivalent to minimizing the forward KL between the optimal and training policies since:

$$\mathcal{L}_{\mathrm{SFT}}(\theta; x) = \sum\nolimits_y -\pi^*(y \,|\, x) \log \pi_\theta(y \,|\, x) = \mathrm{KL}\big[\pi^*(\cdot \,|\, x) \,||\, \pi_\theta(\cdot \,|\, x)\big] + \mathcal{H}(\pi^*(\cdot \,|\, x))\,,$$

where $\mathcal{H}(\pi^*(\cdot \,|\, x))$ is the entropy of $\pi^*(\cdot \,|\, x)$, which does not depend on π_θ.

RL as reverse KL minimization (mode-seeking). The optimal policy for the KL-regularized RL objective (§2.1) is given by $\pi^*(y \,|\, x) = \frac{1}{Z(x)} \pi_{\theta_0}(y \,|\, x) \exp(r(x,y)/\beta)$ (Korbak et al., 2022), where π_{θ_0} is the initial policy, $Z(x) := \sum_y \pi_{\theta_0}(y \,|\, x) \exp(r(x,y)/\beta)$ is the partition function, and $\beta > 0$ is the KL regularization coefficient. This implies that one can view the maximization of the RL objective as minimization of the reverse KL from π^* since:

$$\begin{aligned} J_{\mathrm{RL}}(\theta; x) &= \mathbb{E}_{y \sim \pi_\theta(\cdot \,|\, x)}[r(x,y)] - \beta \cdot \mathrm{KL}[\pi_\theta(\cdot \,|\, x) \,||\, \pi_{\theta_0}(\cdot \,|\, x)] \\ &= -\beta \cdot \mathrm{KL}[\pi_\theta(\cdot \,|\, x) \,||\, \pi^*(\cdot \,|\, x)] + \beta \cdot \log Z(x)\,, \end{aligned}$$

where $\log Z(x)$ does not depend on π_θ (*c.f.* Korbak et al. (2022); Tajwar et al. (2024a)).

3.2 Forward KL Forgets Less in a Uni-Modal Setting

In this section, we demonstrate that forward KL (SFT) leads to less forgetting than reverse KL (RL) under a uni-modal training policy. We model the optimal policy as a mixture of two univariate Gaussian distributions to mirror LM fine-tuning: an "old" mode that corresponds to prior knowledge and a "new" mode that represents the target task. As shown below, results in this setting align with the intuition stated at the beginning of the section, by which the mode-covering forward KL should forget less. However, in the next section we show that once the uni-modal training policy is expanded to a multi-modal one, reverse KL causes less forgetting.

Setup. The optimal policy is modeled by an "old" mode representing prior knowledge and a "new" mode representing a target task:

$$\pi^*(y) = \alpha^* \cdot p_{\mathrm{old}}(y; \theta^*_{\mathrm{old}}) + (1 - \alpha^*) \cdot p_{\mathrm{new}}(y; \theta^*_{\mathrm{new}})\,, \tag{1}$$

where $\alpha^* \in (0,1)$ and the distributions p_{old} and p_{new} are univariate Gaussians with means and standard deviations given by $\theta^*_{\mathrm{old}} = (\mu^*_{\mathrm{old}}, \sigma^*_{\mathrm{old}})$ and $\theta^*_{\mathrm{new}} = (\mu^*_{\mathrm{new}}, \sigma^*_{\mathrm{new}})$, respectively. In this section, the training policy π_θ is modeled as a univariate Gaussian with trainable mean μ and standard deviation σ, *i.e.*, $\theta = (\mu, \sigma)$. We define the target task gain and non-target tasks drop as the change in *overlap area*[3] between the training policy and the modes of the optimal policy. Concretely, the overlap area for the old and new modes is defined as:

$$S_{\mathrm{old}}(\theta) := \frac{\int_{-\infty}^{\infty} \min\{\alpha^* p_{\mathrm{old}}(y), \pi_\theta(y)\}\, dy}{\alpha^*}\,,\ S_{\mathrm{new}}(\theta) := \frac{\int_{-\infty}^{\infty} \min\{(1-\alpha^*) p_{\mathrm{new}}(y), \pi_\theta(y)\}\, dy}{1-\alpha^*}. \tag{2}$$

[3]The overlap area can be formulated via the total variation distance; see Appendix A.2.

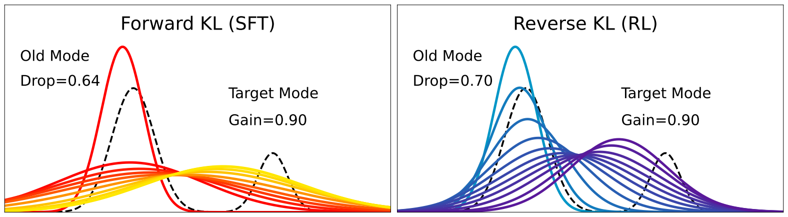

Figure 4: **Forward KL (SFT) with uni-modal training policy forgets less than reverse KL (RL).** Learning and forgetting dynamics of forward KL (left) and reverse KL (right). Dashed lines represent the modes of the optimal policy: p_{old} (left) and p_{new} (right). For forward KL, the data is sampled from the target mode p_{new}; the curve goes from red to yellow as training progresses. For reverse KL, the data is sampled from π_θ; the curve goes from blue to purple. Forgetting corresponds to the decrease of overlap on the left mode and learning a new target task corresponds to the increase in overlap on the right mode.

Notice that $S_{\text{old}}(\theta), S_{\text{new}}(\theta) \in [0,1]$. The target task gain at training step T is accordingly defined by $\Delta_g := S_{\text{new}}(\theta_T) - S_{\text{new}}(\theta_0)$ and the non-target tasks drop is $\Delta_d := S_{\text{old}}(\theta_0) - S_{\text{old}}(\theta_T)$. We initialize the training policy π_θ such that it covers the mode of π^* corresponding to p_{old}, and compare minimizing the forward and reverse KL objectives (defined in §3.1) with respect to p_{new} in terms of their gain-drop tradeoff. The parameters in θ are updated through sample-based gradients, where for forward KL data is sampled from p_{new} and for reverse KL it is sampled from π_θ. See Appendix A.1 for additional implementation details.

Results. Figure 4 shows the forgetting patterns of forward and reverse KL. To reach a target task gain of 0.9, forward KL results in a non-target tasks drop of 0.64 while reverse KL leads to a larger drop of 0.7. This matches common intuition: the mode-covering forward KL stretches the probability mass to cover the new mode while retaining more mass on the old mode compared to the mode-seeking reverse KL. That is, in this setting, forward KL causes less forgetting than reverse KL.

3.3 Reverse KL Forgets Less in a Multi-modal Setting

We showed in §3.2 that the mode-covering properties of forward KL (SFT) lead to less forgetting than reverse KL (RL) when the initial training policy is uni-modal. This stands in contrast to the experiments of §2.3, which show that in practical LM post-training settings, RL is more resilient to forgetting. In this section, we reconcile this discrepancy by showing that when we allow the initial training policy to have multiple modes, arguably a closer match to practice, the mode-seeking reverse KL results in less forgetting.

Setup. We consider the setup of §3.2, where the optimal policy is modeled as a mixture of two Gaussian distributions (Equation 1). Instead of modeling the training policy as a uni-modal Gaussian, we now model it as a bi-modal distribution:

$$\pi_\theta(y) = \alpha \cdot q_{\text{old}}(y;\theta_{\text{old}}) + (1-\alpha) \cdot q_{\text{new}}(y;\theta_{\text{new}}), \tag{3}$$

where $\theta = (\alpha, \theta_{\text{old}}, \theta_{\text{new}})$ is the trainable parameters, with $\alpha \in [0,1]$ being a mixture weighting, $\theta_{\text{old}} = (\mu_{\text{old}}, \sigma_{\text{old}})$ defining the mean and standard deviation of a univariate Gaussian q_{old}, and $\theta_{\text{new}} = (\mu_{\text{new}}, \sigma_{\text{new}})$ similarly defining a univariate Gaussian q_{new}. We initialize the training policy π_θ such that q_{old} roughly covers the mode of π^* corresponding to p_{old} and, as in §3.2, compare the gain-drop tradeoffs exhibited by forward and reverse KL minimization with respect to p_{new}. See Appendix A.1 for additional implementation details.

Results. Figure 5 shows that, with a high learning rate, achieving a target task gain of 0.9 with forward KL causes severe forgetting—the area overlap with p_{old} drops by 0.12. Reducing the learning rate can mitigate forgetting of the old mode, but leads to failure in learning the target p_{new}. By contrast, reverse KL shifts q_{new} toward p_{new} while largely keeping the old mode intact. This simulation demonstrates that, for bi-modal policies, reverse KL can match a new target mode without redistributing probability mass from a mode that represents prior knowledge.

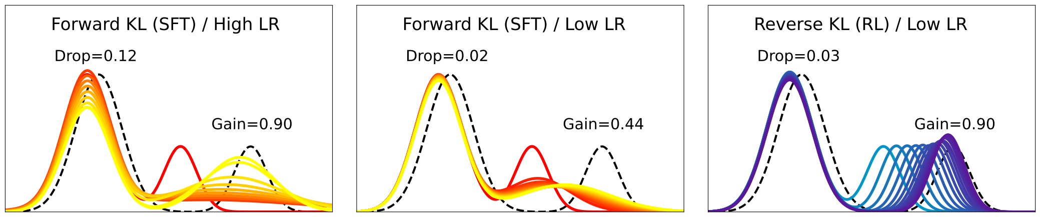

Figure 5: **Reverse KL (RL) with multi-modal training policy forgets less than forward KL (SFT).** Learning and forgetting patterns of forward KL with different high (0.15) and low (0.01) learning rates (left and middle) and reverse KL (right). Dashed lines represent the modes of the optimal policy: p_{old} (left) and p_{new} (right). For forward KL, the data is sampled from the target mode p_{new}; the curve goes from red to yellow as training progresses. For reverse KL, the data is sampled from π_θ; the curve goes from blue to purple. Forgetting corresponds to the decrease of overlap on the left mode and learning a new task is the increase in overlap on the right mode.

4 Learning from On-Policy Data Mitigates Forgetting

The experiments of §2 demonstrated that RL causes less forgetting than SFT. By considering a simplified setting in §3, we identified that the mode-seeking behavior of RL, which stems from its usage of on-policy data, may underlie its robustness to forgetting. We now verify this prospect by demonstrating in practical settings that the robustness of RL to forgetting indeed arises from its use of on-policy data, as opposed to other algorithmic choices such as KL regularization or an advantage estimator (§4.1). We then explore the following natural question: *what degree of on-policy data allows mitigating forgetting?* As evident from the results of Self-SFT in Figure 2, generating data only from the initial policy is not enough. However, we show that SFT with approximately on-policy data, generated at every epoch or with on-policy traces produced by RL, can suffice for substantially reducing forgetting (§4.2). This highlights the potential of mitigating forgetting through approximately on-policy data, which can be substantially more efficient to obtain than fully on-policy data.

4.1 On-Policy Data is the Primary Contributor for Mitigating Forgetting

There are three distinctions between RL, as implemented via GRPO, and SFT (*cf.* §2.1): *(i)* RL trains on on-policy data, generated by the current policy, while SFT uses off-policy data; *(ii)* the RL objective typically includes KL regularization with respect to the initial policy while SFT does not; and *(iii)* RL multiplies gradients of responses by an advantage estimate while SFT does not. Below, we identify on-policy data (*i.e.*, *(i)*) as the source of RL's robustness to forgetting by ruling out the necessity of *(ii)* and *(iii)*. We note that our results stand in contrast to Lai et al. (2025), which hypothesized that a particular form of advantage estimation mitigates forgetting.

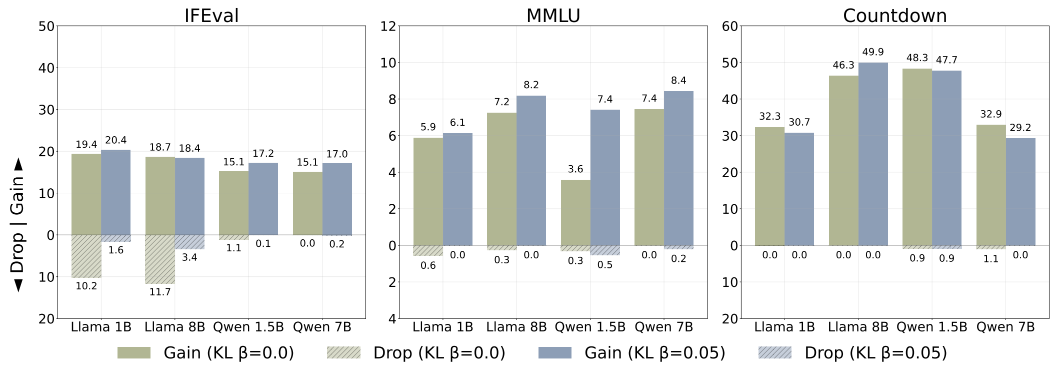

Figure 6: **KL regularization is not a major contributor to RL's lesser degree of forgetting.** Comparison of GRPO with KL regularization ($\beta = 0.05$) and without it ($\beta = 0.0$). Except for the Llama model family and IFEval target task, the non-regularized GRPO achieves a similar gain-drop tradeoff as regularized GRPO.

		IFEval		MMLU		Countdown	
		Gain (%) ↑	Drop ↓ (%)	Gain (%) ↑	Drop (%) ↓	Gain (%) ↑	Drop (%) ↓
Llama 3.1 8B Inst.	SFT	25.2	27.8	11.1	38.5	25.5	36.4
	REINFORCE	17.8	7.7	8.6	-0.1	7.5	-0.8
	GRPO	18.4	3.4	14.6	-0.2	60.4	-0.5
Qwen 2.5 7B Inst.	SFT	24.2	5.6	9.4	14.6	10.4	29.2
	REINFORCE	5.7	2.9	6.4	-0.6	11.9	-0.1
	GRPO	17.0	0.2	8.4	0.2	29.2	-0.3

Table 1: **REINFORCE also enjoys a lesser degree of forgetting as GRPO.** Gain and drop comparison of SFT, REINFORCE, and GRPO. The SFT and GRPO results are taken from Figure 2.

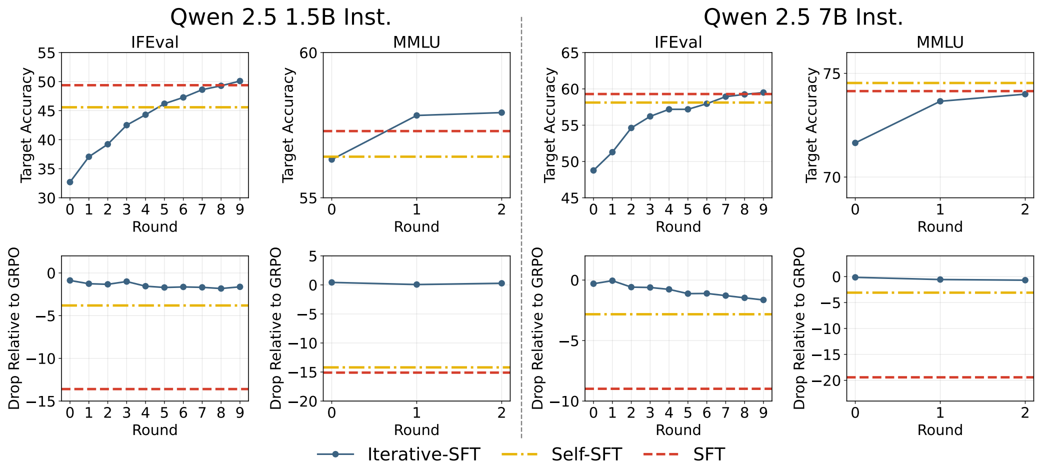

Figure 7: **Approximately on-policy data can suffice for mitigating forgetting in SFT.** This figure compares three SFT variants using Qwen 2.5 1.5B Instruct and Qwen 2.5 7B Instruct on IFEval and MMLU. Top rows: target task accuracy. Bottom rows: average non-target accuracy drop relative to the drop of GRPO (i.e., $\Delta_d - \Delta_d^{\mathrm{GRPO}}$). The three SFT variants include: *(1)* Iterative-SFT, which uses data generated at the start of each round (*i.e.*, epoch); *(2)* Self-SFT, which uses data generated from the initial policy; and *(3)* SFT, which uses fully off-policy data generated by a separate expert model (Llama-3.3-70B-Instruct). While SFT and Self-SFT suffer from severe forgetting, our results show that using approximately on-policy data, generated at the start of each epoch, can suffice for mitigating forgetting (Iterative-SFT).

KL regularization does not explain robustness to forgetting. KL regularization is commonly applied during RL to prevent the policy from drifting too far from its initialization (Ouyang et al., 2022; Shao et al., 2024). We examine whether this regularization accounts for the lesser forgetting of RL. As Figure 6 shows, non-regularized GRPO achieves a similar target task gain and non-target tasks drop tradeoff as KL-regularized GRPO across all considered models and datasets, except for Llama models trained on IFEval. These results suggest that the use of KL regularization does not underlie the robustness of RL to forgetting.

REINFORCE can be as robust as GRPO to forgetting. We compare GRPO with REINFORCE (Williams, 1992), a classical policy gradient RL algorithm that does not employ an advantage estimator. Table 1 shows that REINFORCE lags behind GRPO in optimizing the target task accuracy, yet maintains a similar low level of forgetting. This suggests that algorithmic differences, such as the advantage estimator used in RL, primarily affect the magnitude of performance gains, whereas the mitigation of forgetting can be primarily attributed to the use of on-policy data.

4.2 Approximately On-Policy Data Can Suffice for Mitigating Forgetting

We have identified the usage of on-policy data as the main contributor for the robustness of RL to forgetting. However, generating on-policy data at each step entails a non-negligible compute overhead. We therefore explore the degree of "on-policyness" required to enjoy the benefit of reduced forgetting. Figure 2 showed that Self-SFT, which generates data only from the initial policy, suffers from severe forgetting. On the other hand, RL, which generates data at every step and thus represents the most on-policy end of the spectrum, is robust to forgetting. We now test whether Iterative-SFT, an *approximately* on-policy approach that iteratively trains on data generated at the start of each epoch (Zelikman et al., 2022; Dong et al., 2023; Xiong et al., 2025), can suffice for mitigating forgetting.

Figure 7 compares the target task accuracy and the drop in non-target tasks relative to GRPO of Iterative-SFT, Self-SFT, and SFT. We find that Iterative-SFT is able to reach a target accuracy that is higher than or comparable to that of SFT, while only exhibiting mild to no forgetting. We also test an additional approximately on-policy approach that applies SFT on data generated during an RL run, and observe reduced forgetting as well (see Appendix A.4.1). Overall, these results highlight that while RL remains most effective in forgetting mitigation, making SFT more on-policy or directly applying SFT on RL data can suffice for reducing forgetting.

5 Related Work

Catastrophic forgetting. Catastrophic forgetting has been studied since early research on connectionist models (McCloskey & Cohen, 1989). Early studies mostly focused on preventing parameters from drastically changing (Kirkpatrick et al., 2017; Li & Hoiem, 2018; Lopez-Paz & Ranzato, 2017). In the context of LM post-training, patterns of catastrophic forgetting differ due to the massive data used during pre-training (Luo et al., 2023; Shi et al., 2024; Wu et al., 2024). While LMs typically do not drastically forget all pre-trained knowledge, post-training LMs often leads to degradation in performance, which has been called "alignment tax" (Bai et al., 2022; Ouyang et al., 2022). Prior work found more severe forgetting when considering contrasting domains such as instruction following and safety (Qi et al., 2023; He et al., 2024). Though, Lee et al. (2024); Chen et al. (2024) suggest that forgotten behaviors or abilities can be revived with little re-training. Kotha et al. (2024) posited that forgetting happens when the LM infers a wrong mode from the mixture of distributions to perform the task, and a carefully selected prompt can recover forgetting. Our work draws inspiration from the mixture of distributions perspective to establish intuition.

LM post-training. Post-training techniques such as SFT and RL are widely used for endowing pre-trained LMs with desired behaviors or enhancements in target domains. SFT relies on the ground truth responses provided in the datasets. RL allows the model to generate its own responses and only provide a reward at the end, be it from a parameterized reward model (Schulman et al., 2017) or a verifiable reward module (Shao et al., 2024; Lambert et al., 2024). Recent studies have shown that SFT and RL exhibit distinct characteristics. Chu et al. (2025) showed that RL is able to generalize to unseen distributions while SFT mostly memorizes. (Wang et al., 2025) observed that RL can benefit from even training on a single example without severe overfitting. (Mukherjee et al., 2025) reported that RL naturally fine-tunes a smaller network compared to SFT. A common thread connecting these observations is that the parameter update during RL training is more local and targeted. Other methods such as RAFT (Dong et al., 2023) and STaR (Zelikman et al., 2022) performs SFT in several rounds and can be viewed as on-policy offline RL. This paper complements these studies and provide a forgetting-centric view on the difference between SFT and RL.

Concurrent work. Similarly to our work, Lai et al. (2025); Shenfeld et al. (2025) have concurrently found that RL exhibits less forgetting than SFT. However, Lai et al. (2025) attribute RL's robustness to an implicit regularization of a particular advantage estimator. We provide evidence against this claim in §4, and instead identify the crucial role of on-policy data in mitigating forgetting. Shenfeld et al. (2025) also highlight the benefits of on-policy data through a perspective that is complementary to ours (§3). Despite the similar observations, we find that their hypothesis on the connection between KL divergence from the initial policy and forgetting does not always hold in our setting (see Appendix A.5). Moreover, our work goes beyond fully on-policy data and demonstrates the potential of approximately on-policy data in more efficiently mitigating forgetting.

6 Conclusion

We systematically compared catastrophic forgetting in SFT and RL for LM post-training. Across tasks, scales, and model families, we found that RL consistently achieves strong target performance with substantially less forgetting than SFT. Our experiments in both simplified and real-world settings establish that the robustness of RL to forgetting primarily stems from its use of on-policy data, rather than other algorithmic choices such as the advantage estimate or KL regularization. Furthermore, they highlight the potential of efficiently mitigating forgetting by incorporating approximately on-policy data, sampled asynchronously or at the start of each epoch.

Limitations and future directions. Our work provides evidence that RL is more robust than SFT to forgetting across several tasks, model families, and scales. However, investigating how forgetting patterns vary as the model and dataset sizes are further scaled, beyond our compute budget, remains a valuable direction for future work. Moreover, while we provide intuition for why RL forgets less than SFT based on a simplified mixture-of-Gaussians setting (§3) and empirically identify the use of on-policy data as a main cause for this difference in forgetting (§4), additional research is necessary to theoretically establish the role of on-policy data in mitigating forgetting. Going forward, the issue of forgetting becomes central as the community moves toward building agents that continually learn from experience (Silver & Sutton, 2025). Deciding what data to consume is consequential to the stability of the agent. Our results indicate that data generated on-policy will better preserve existing capabilities, and is therefore safer to learn from, than off-policy data such as content on the internet or experience from other agents. In a similar vein, our insights lays groundwork for understanding forgetting in the emerging paradigm of test-time training (Sun et al., 2020; Hardt & Sun, 2024).

7 Acknowledgement

We thank the members of Princeton Language Intelligence Group for providing comments on the manuscript. We also thank Kevin Lu from Thinking Machines Lab for the valuable feedback. This research is supported by the National Science Foundation (IIS-2211779) and is also supported in part by Schmidt Sciences. NR is supported in part by the Zuckerman STEM Leadership Program.

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. URL `https://arxiv.org/abs/2204.05862`.

Alan Chan, Hugo Silva, Sungsu Lim, Tadashi Kozuno, A. Rupam Mahmood, and Martha White. Greedification operators for policy optimization: Investigating forward and reverse kl divergences. *Journal of Machine Learning Research*, 23:1–79, 2022. URL `https://www.jmlr.org/papers/v23/21-054.html`.

Howard Chen, Jiayi Geng, Adithya Bhaskar, Dan Friedman, and Danqi Chen. Continual memorization of factoids in large language models. *arXiv preprint arXiv:2411.07175*, 2024. URL `https://arxiv.org/abs/2411.07175`.

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 *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025. URL `https://arxiv.org/abs/2501.17161`.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang,

R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. URL `https://arxiv.org/abs/2501.12948`.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, KaShun Shum, and Tong Zhang. RAFT: Reward ranked finetuning for generative foundation model alignment. *Transactions on Machine Learning Research*, 2023. URL `https://arxiv.org/abs/2304.06767`.

Aaron Grattafiori et al. The llama 3 herd of models. *arXiv*, 2024. URL `https://arxiv.org/abs/2407.21783`.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024) — Datasets & Benchmarks Track*, 2024. URL `https://proceedings.neurips.cc/paper_files/paper/2024/hash/0f69b4b96a46f284b726fbd70f74fb3b-Abstract-Datasets_and_Benchmarks_Track.html`.

Moritz Hardt and Yu Sun. Test-time training on nearest neighbors for large language models. In *International Conference on Learning Representations (ICLR)*, 2024. URL `https://openreview.net/forum?id=CNL2bku4ra`.

Luxi He, Mengzhou Xia, and Peter Henderson. What's in your "safe" data?: Identifying benign data that breaks safety. In *First Conference on Language Modeling (COLM)*, 2024. URL `https://arxiv.org/abs/2404.01099`.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021a. URL `https://arxiv.org/abs/2009.03300`.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *Proceedings of the Neural Information Processing Systems (NeurIPS) 2021, Datasets & Benchmarks Track*, 2021b.

Liwei Jiang, Kavel Rao, Seungju Han, Allyson Ettinger, Faeze Brahman, Sachin Kumar, Niloofar Mireshghallah, Ximing Lu, Maarten Sap, Yejin Choi, and Nouha Dziri. Wildteaming at scale: From in-the-wild jailbreaks to (adversarially) safer language models. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024) — Datasets & Benchmarks Track*, 2024. URL `https://proceedings.neurips.cc/paper_files/paper/2024/file/54024fca0cef9911be36319e622cde38-Paper-Conference.pdf`.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Katherine Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwińska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences (PNAS)*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL `https://www.pnas.org/doi/10.1073/pnas.1611835114`.

Tomasz Korbak, Hady Elsahar, Germán Kruszewski, and Marc Dymetman. On reinforcement learning and distribution matching for fine-tuning language models with no catastrophic forgetting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. URL `https://proceedings.neurips.cc/paper_files/paper/2022/file/67496dfa96afddab795530cc7c69b57a-Paper-Conference.pdf`.

Suhas Kotha, Jacob Mitchell Springer, and Aditi Raghunathan. Understanding catastrophic forgetting in language models via implicit inference. In *International Conference on Learning Representations (ICLR)*, 2024. URL `https://arxiv.org/abs/2309.10105`.

Song Lai, Haohan Zhao, Rong Feng, Changyi Ma, Wenzhuo Liu, Hongbo Zhao, Xi Lin, Dong Yi, Min Xie, Qingfu Zhang, Hongbin Liu, Gaofeng Meng, and Fei Zhu. Reinforcement fine-tuning naturally mitigates forgetting in continual post-training. *arXiv preprint arXiv:2507.05386*, 2025. URL `https://arxiv.org/abs/2507.05386`.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024. URL `https://arxiv.org/abs/2411.15124`.

Andrew Lee, Xiaoyan Bai, Itamar Pres, Martin Wattenberg, Jonathan K. Kummerfeld, and Rada Mihalcea. A mechanistic understanding of alignment algorithms: A case study on dpo and toxicity. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, pp. 26361–26378. PMLR, 2024. URL `https://arxiv.org/abs/2401.01967`.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, December 2018. doi: 10.1109/TPAMI.2017.2773081. URL `https://doi.org/10.1109/TPAMI.2017.2773081`.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pp. 6467–6476, Long Beach, CA, USA, 2017. Curran Associates, Inc. URL `https://papers.nips.cc/paper_files/paper/2017/file/f87522788a2be2d171666752f97ddebb-Paper.pdf`.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. In *arXiv preprint 2308.08747*, 2023.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower (ed.), *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Academic Press, San Diego, CA, 1989. doi: 10.1016/S0079-7421(08)60536-8. URL `https://doi.org/10.1016/S0079-7421(08)60536-8`.

Sagnik Mukherjee, Lifan Yuan, Dilek Hakkani-Tur, and Hao Peng. Reinforcement learning finetunes small subnetworks in large language models. *arXiv preprint arXiv:2505.11711*, 2025. URL `https://arxiv.org/abs/2505.11711`.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. URL `https://arxiv.org/abs/2203.02155`.

Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. Tinyzero. https://github.com/Jiayi-Pan/TinyZero, 2025. Accessed: 2025-01-24.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! *arXiv preprint arXiv:2310.03693*, 2023. URL `https://arxiv.org/abs/2310.03693`.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL `https://openreview.net/forum?id=Xaf289hqmZ`.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL `https://arxiv.org/abs/1707.06347`.

Shuai Shao, Yifan Liang, Zihan Ding, Yichong Wang, Zichen Li, Boya Zhao, Weilin Jin, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL `https://arxiv.org/abs/2402.03300`.

Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. Rl's razor: Why online reinforcement learning forgets less. *arXiv preprint arXiv:2509.04259*, 2025. URL `https://arxiv.org/abs/2509.04259`.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, and Hao Wang. Continual learning of large language models: A comprehensive survey. *arXiv preprint arXiv:2404.16789*, 2024. URL `https://arxiv.org/abs/2404.16789`.

David Silver and Richard S. Sutton. Welcome to the era of experience. *Preprint (to appear in *Designing an Intelligence*, MIT Press)*, 2025. URL `https://storage.googleapis.com/deepmind-media/Era-of-Experience%20/The%20Era%20of%20Experience%20Paper.pdf`.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A. Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119, pp. 9229–9248. PMLR, 2020. URL `https://proceedings.mlr.press/v119/sun20b.html`.

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. Preference fine-tuning of LLMs should leverage suboptimal, on-policy data. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024a. URL `https://openreview.net/forum?id=RFSlmeFQ9T`.

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. Preference fine-tuning of LLMs should leverage suboptimal, on-policy data. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 47441–47474. PMLR, Jul 2024b. URL `https://proceedings.mlr.press/v235/tajwar24a.html`.

Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Lucas Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, Weizhu Chen, Shuohang Wang, Simon Shaolei Du, and Yelong Shen. Reinforcement learning for reasoning in large language models with one training example. *arXiv preprint arXiv:2504.20571*, 2025. URL `https://arxiv.org/abs/2504.20571`.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992. doi: 10.1007/BF00992696. URL `https://doi.org/10.1007/BF00992696`.

Tongtong Wu, Linhao Luo, Yuan-Fang Li, Shirui Pan, Thuy-Trang Vu, and Gholamreza Haffari. Continual learning for large language models: A survey. *arXiv preprint arXiv:2402.01364*, 2024. URL `https://arxiv.org/abs/2402.01364`.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, and Hanze Dong. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*, 2025. URL `https://arxiv.org/abs/2504.11343`.

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

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. 29144–29157, 2022. URL `https://arxiv.org/abs/2203.14465`.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023. URL `https://arxiv.org/abs/2311.07911`.

A Additional Experiments and Implementation Details

A.1 Simulation Details

Setup. We use a univariate mixture-of-Gaussians synthetic task to compare forward KL (FKL; SFT analogue) update against reverse KL (RKL; RL analogue). We calculate gradient updates using $n = 1000$ samples. For evaluation and plots, densities are computed on a uniform grid at every 100 iterations.

Single-mode setting. We run the gradient step updates for $T = 1000$ iterations or when the target task gain reaches 0.9. The training policy starts as a single-mode univariate Gaussian at the old mode, initialized as $\mathcal{N}(-3.2, 1.0)$ (75% old mass), and is adapted toward the same target mixture used above: $0.75 \cdot \mathcal{N}(-3.0, 1.0) + 0.25 \cdot \mathcal{N}(3.5, 0.7)$. We use an FKL learning rate 0.05 and a RKL learning rate 0.05.

Bi-modal setting. We run the gradient step updates for $T = 1000$ iterations or when the target task gain reaches 0.9. The initial policy $\pi_\theta(x)$ is a two-component mixture with weight 0.75 on an "old" Gaussian $\mathcal{N}(-3.5, 1.0)$ and 0.25 on a "new" Gaussian $\mathcal{N}(0.5, 0.7)$. The target $\pi^*(x)$ is a mixture with the same weights over $\mathcal{N}(-3.0, 1.0)$ (old) and $\mathcal{N}(3.5, 0.7)$ (new). We sweep two FKL learning rates $\{0.15,\ 0.01\}$ and use a RKL learning rate 0.01.

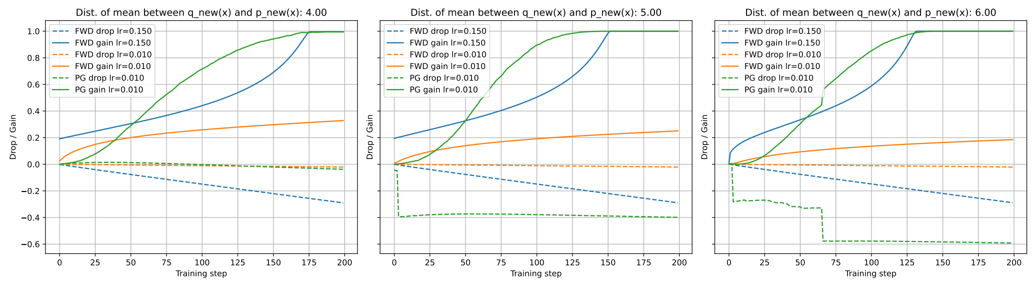

Figure 8: Simulation comparison with different distances ($[4.0, 5.0, 6.0]$) between p_{new} and q_{new}.

RL forgets when the target distribution is far. We show in Figure 8 the simulation results with varying distance ($4.0 - 6.0$) between q_{new} and p_{new}. We observe that as the distance gets larger, RL begins to suffer from forgetting as well. suggesting that RL is not immune to forgetting when the target task is drastically far away from the starting modes.

A.2 Connection Between Area Overlap and Total Variation Distance

Let $f, g : \mathcal{D} \to \mathbb{R}_{\geq 0}$ be, possibly unnormalized, integrable density functions over a domain $\mathcal{D}$. The total variation (TV) between f and g is defined by:

$$\mathrm{TV}(f, g) \ := \ \frac{1}{2} \int_{\mathcal{D}} |f(y) - g(y)| dy \,.$$

Notice that:

$$\begin{aligned} \int_{\mathcal{D}} \min\{f(y), g(y)\}\, dy &= \int_{\mathcal{D}} \frac{1}{2} \big(f(y) + g(y) - |f(y) - g(y)|\big) dy \\ &= \frac{1}{2}\left(\int_{\mathcal{D}} f(y)dy + \int_{\mathcal{D}} g(y)dy\right) - \frac{1}{2}\int_{\mathcal{D}} |f(y) - g(y)| dy \\ &= \frac{1}{2}\left(\int_{\mathcal{D}} f(y)dy + \int_{\mathcal{D}} g(y)dy\right) - \mathrm{TV}(f, g)\,. \end{aligned} \tag{4}$$

Now, in the context of §3.2, recall that the *area overlap* of the training policy π_θ with respect to the old mode of the optimal policy is defined by (Equation (2)):

$$S_{\text{old}}(\theta) = \frac{\int_{-\infty}^{\infty} \min\{\alpha^* p_{\text{old}}(y), \pi_\theta(y)\}\, dy}{\alpha^*}\,.$$

Choosing $f = \alpha^* p_{\text{old}}$ and $g = \pi_\theta$, by Equation (4) we may write $S_{\text{old}}(\theta)$ as follows:

$$S_{\text{old}}(\theta) = \frac{\frac{1}{2}(\alpha^* + 1) - \text{TV}\big(\alpha^* p_{\text{old}}, \pi_\theta\big)}{\alpha^*} = \frac{1}{2} + \frac{1}{2\alpha^*} - \frac{1}{\alpha^*}\text{TV}\big(\alpha^* p_{\text{old}}, \pi_\theta\big)\,.$$

Hence, the non-target tasks drop at training step T is equal to the normalized increase in total variation distance between the training policy and the (scaled) old component of the optimal policy:

$$\Delta_d = S_{\text{old}}(\theta_0) - S_{\text{old}}(\theta_T) = \frac{\text{TV}\big(\alpha^* p_{\text{old}}, \pi_{\theta_T}\big) - \text{TV}\big(\alpha^* p_{\text{old}}, \pi_{\theta_0}\big)}{\alpha^*}\,.$$

Similarly, the area overlap of the training policy π_θ with respect to the new mode of the optimal policy is given by:

$$S_{\text{new}}(\theta) = \frac{\frac{1}{2}(\alpha^* + 1) - \text{TV}\big(\alpha^* p_{\text{old}}, \pi_\theta\big)}{\alpha^*} = \frac{1}{2} + \frac{1}{2(1-\alpha^*)} - \frac{1}{1-\alpha^*}\text{TV}\big((1-\alpha^*) p_{\text{new}}, \pi_\theta\big)\,.$$

This implies that the target task gain at training step T is equal to the normalized decrease in total variation distance between the training policy and the (scaled) new component of the optimal policy:

$$\Delta_g = S_{\text{new}}(\theta_T) - S_{\text{new}}(\theta_0) = \frac{\text{TV}\big((1-\alpha^*) p_{\text{new}}, \pi_{\theta_T}\big) - \text{TV}\big((1-\alpha^*) p_{\text{new}}, \pi_{\theta_0}\big)}{1-\alpha^*}\,.$$

A.3 Implementation Details

Training details. We used the AdamW optimizer. The learning rate was initialized to $1e{-}4$ for Llama-3.2-1B-Instruct and Qwen-2.5-1.5B-Instruct and $5e{-}6$ for Llama-3.1-8B-Instruct and Qwen-2.5-7B-Instruct. We use cosine scheduler with warp-up step ratio 0.03 over the course of training. Each model was trained with a batch size of 128 for IFEval and MMLU and 64 for Countdown. Unless otherwise specified, training was run for 2 epochs.

For SFT, we minimized the cross-entropy loss with a maximum sequence length of 4096. For Self-SFT, we generate 5 responses from the initial model and filter out the incorrect responses based on the reward model. We keep all the correct responses of each prompt into the dataset. For RL experiments, we used the GRPO algorithm with a KL-penalty coefficient of 0.05 and apply updates right after the group samples are generated (hence no advantage clipping). We generate a group size of 5 for each prompt. All experiments were implemented in PyTorch and trained on maximally 8 H100 GPUs with mixed-precision (bfloat16) training.

Data. For IFEval (Zhou et al., 2023), we use the dataset provided in the Tulu3 for post-training Lambert et al. (2024). We split the dataset into the training set containing $13,000$ and the evaluation set of $1,972$ examples. For MMLU, we split the dataset Hendrycks et al. (2021a) into the training set containing $12,000$ examples and the evaluation set containing $2,042$ examples. For Countdown, we generate data following the procedure in Pan et al. (2025). We split the dataset into the training set containing $10,000$ examples and the evaluation set contains $1,000$ examples.

Prompts. Throughout our experiments, we use the following chat format:

```
User:
{prompt}

Assistant:
{response}
```

For SFT, the target response is provided; for RL, the response is generated by the training policy at each optimization step.

IFEval: we use the prompt in the original dataset exactly.

MMLU: the prompt is:

~~~
```{question}
Answer options:
A. {option_A}
B. {option_B}
C. {option_C}
D. {option_D}

Reason about it and answer with "The answer is:  <option>"```
~~~

, where the question and options are provided in the MMLU dataset.

Countdown: we use the following prompt `"In this task, you need to use a list of numbers x = {x} to create an equation that leads up to the target number y = {y} using the basic arithmetic operations (+, -, *, /), and each number can only be used once. Think and return the final answer in $\\boxed{ }$."` , where {x} is a list of integers and {y} is the target integer.

A.4 Extra Experiments and Ablations

A.4.1 SFT Using RL Traces

Data generated by RL throughout training is on-policy with regard to the model at each optimization step. When this RL data is later used for SFT, the process moves away from being fully on-policy, though it remains distinct from fully off-policy approaches such as SFT. We investigate whether SFT on RL data can also mitigate forgetting. In Figure 9, we observe that SFT trained on RL (GRPO) data trails full RL marginally in terms of gains but exhibits only slightly larger forgetting. This highlights a yet-to-be-identified benefit of using RL data for SFT (DeepSeek-AI et al., 2025).

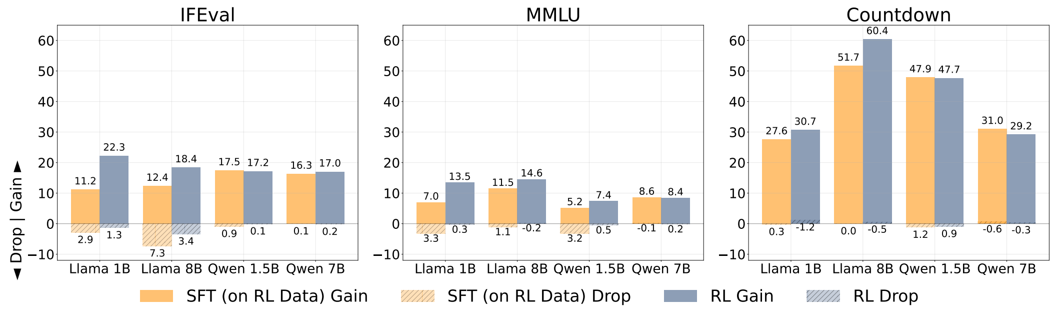

Figure 9: **SFT over on-policy traces produced by RL exhibits reduced forgetting.** This plot shows the comparison between SFT trained on RL (GRPO) data and RL (GRPO).

A.5 KL Divergence From Initial Policy and Forgetting Correlate Moderately

		IFEval		MMLU		Countdown	
		Drop (%)	KL	Drop (%)	KL	Drop (%)	KL
Llama-3.2-1B-Instruct	Self-SFT	6.9	52.4	34.6	39.3	25.3	796.7
	SFT	26.2	61.1	28.9	48.4	24.2	1254.7
	GRPO	1.6	2.6	0.3	4.1	-0.6	70.6
Qwen-2.5-1.5B-Instruct	Self-SFT	3.0	25.0	14.0	3.0	19.5	896.1
	SFT	6.2	47.8	11.9	9.2	29.5	846.6
	GRPO	0.6	1.5	0.5	0.4	0.9	34.9

Table 2: Non-target tasks drop and $\mathrm{KL}[\pi_{\theta_0} \,||\, \pi_\theta]$ for the Llama-3.2-1B-Instruct and Qwen-2.5-1.5B-Instruct models from Figure 2.

Shenfeld et al. (2025) have concurrently found that RL exhibits less forgetting than SFT, attributing RL's robustness to an implicit regularization of RL toward policies with low KL divergence from the initial policy π_{θ_0}. In particular, Shenfeld et al. (2025) empirically identify $\mathrm{KL}[\, \pi_{\theta_0} \,||\, \pi_\theta \,]$ as an indicator for the extent of forgetting, mostly through an extensive evaluation on synthetic tasks. We explore the connection between $\mathrm{KL}[\, \pi_{\theta_0} \,||\, \pi_\theta \,]$ and forgetting in the setup of §2, where the KL divergence is estimated based on 100 examples from the evaluation set. Table 2 shows that, in accordance with the hypothesis of Shenfeld et al. (2025), GRPO exhibits both smaller KL divergences and smaller drops in non-target tasks performance compared to the SFT variants. Furthermore, the Pearson correlation between KL divergence and non-target tasks drop across all models, methods, and datasets is 0.52. Yet, when comparing Self-SFT and SFT, the relation between KL divergence and forgetting is less monotonic—a larger KL does not necessarily imply a higher degree of forgetting. This indicates that the relationship between KL divergence and forgetting may be more subtle than Shenfeld et al. (2025) suggest.