Learning to Remember: How Early Data Exposure Protects Post-Trained Capabilities from Catastrophic Forgetting

Lawrence Feng

April 2026

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Committee

Aditi Raghunathan (Advisor)

Submitted in partial fulfillment of the requirements
for the Senior Honors Thesis

Keywords: language models, continual learning, catastrophic forgetting, post-training, pretraining data composition, data mixing, replay, dropout regularization, low-rank adaptation

Abstract

How can we train models whose post-trained capabilities survive subsequent fine-tuning? Rather than focusing on downstream interventions to mitigate forgetting of upstream capabilities, we study how upstream training choices — that is, the manner in which a capability is acquired — shape how robustly that capability is retained. We investigate this question in a controlled three-stage language-model pipeline: pretraining, post-training to acquire a target capability, and downstream fine-tuning on a new objective. Across 135M and 1B models, two post-training domains, and two downstream fine-tuning tasks, we find that immediate post-training performance does not reliably predict retention after subsequent fine-tuning: training recipes that look equivalent immediately after post-training can retain the target capability very differently after subsequent fine-tuning. In particular, *early exposure* — mixing post-training data into pretraining — consistently improves the frontier between retained upstream performance and downstream performance. In compute-matched experiments, where the target data must be allocated between pretraining and post-training, we find that the optimum lies at neither extreme. Together with our other empirical and theoretical findings, this supports the view that post-training drives immediate specialization while early exposure improves robustness to later forgetting. Replay and dropout, typically used to mitigate forgetting as it occurs during fine-tuning, provide complementary gains to early exposure when applied during post-training. Our findings suggest that robustness to subsequent fine-tuning should be treated as a first-class objective of upstream training, addressed preventatively through choices like early exposure rather than reactively during fine-tuning itself.

1 Introduction

When a post-trained language model is released for downstream fine-tuning, its carefully acquired capabilities are at risk. Fine-tuning on a new objective routinely causes catastrophic forgetting of behaviors introduced during post-training — whether instruction following, domain knowledge, coding ability, or safety-related behavior (Yang et al., 2025; Olmo et al., 2025).

Most prior work treats this as a problem for the downstream fine-tuner to solve. If fine-tuning degrades prior capabilities, the natural response is to modify that fine-tuning stage: replay earlier data (Bethune et al., 2025; Kotha & Liang, 2026), regularize the update (Kirkpatrick et al., 2017), restrict the trainable parameters (Hu et al., 2021; Biderman et al., 2024), or jointly optimize competing objectives (Wortsman et al., 2022a;b).

We take a complementary view: robustness to subsequent fine-tuning should be treated as an objective of upstream model development. Upstream developers typically train in two stages — first on a large general corpus to build broad language understanding, then on a smaller, often scarce, targeted dataset to instill specific capabilities. Because this second stage uses limited data, how and when it is used matters. Our central intuition is that how a model learns a capability shapes how robustly it is retained: two models that reach identical post-training performance can differ substantially in how well those capabilities survive later adaptation.

To study this question, we use a controlled three-stage pipeline reflecting this standard practice: an upstream developer first pretrains on a broad corpus, then post-trains on a smaller targeted dataset to acquire specific capabilities, and finally hands the resulting model to a downstream user who fine-tunes it on a new objective (Figure 1). We study this framework across multiple controlled settings spanning different post-training and downstream fine-tuning regimes (Table 1), including both domain and behavioral adaptation, and evaluate these settings for 135M parameter models, extending our findings to 1B parameter models. We hold the downstream fine-tuning method fixed, applying standard supervised fine-tuning, and sweep its learning rate to characterize how upstream choices shape the tradeoff between downstream performance, retention of the post-trained capability, and performance on the broader pretraining distribution. Accordingly, our evaluation centers on the tradeoff frontier induced by different methods, rather than immediate post-training performance alone.

Our main intervention is simple: we expose the model to some of the eventual post-training data earlier by mixing it in during pretraining. Across datasets and model sizes, we find that this early exposure improves the tradeoff between retained upstream capability and downstream fine-tuning loss (Figure 3), even when it has little or no visible effect on immediate post-training performance (Section 4.1).

Figure 1: Overview of our three-stage experimental setup, in contrast to a typical two-stage setup. A first party pretrains then post-trains a model with the goal of achieving high performance on domain X. Subsequently, downstream users fine-tune θ_{post} for a task Y, causing catastrophic forgetting of domain X. Previous work investigates interventions in the third stage: how can we fine tune for Y while mitigating forgetting on X. In this work, we investigate how *the way in which* X is learned affects how it is (or not) forgotten.

Why does early exposure help? Our theoretical account (Section 5) suggests that mixing during pretraining allows the post-training capability to be represented in specialized features that are less vulnerable to subsequent interference. Our compute-matched experiments reinforce this view: even under a fixed budget of post-training data, the optimum does not lie at either extreme of allocating all exposure to pretraining or all exposure to post-training, but between the two (Section 4.2).

If the manner of learning matters, a natural follow-up question arises: what other interventions can shape how a capability is acquired? We study replay and dropout from this perspective (Section 4.4). Intuitively, replay encourages the model to acquire the new domain without overwriting prior features, while dropout promotes more distributed and robust representations. Importantly, we evaluate whether the benefits of these interventions persist after a later downstream fine-tuning stage. We find that both improve the tradeoff frontier while remaining complementary to pretraining-time mixing.

Together, our findings show that the upstream training process is a meaningful lever for shaping robustness to subsequent fine-tuning. In particular, a remarkably simple intervention—mixing a small amount of post-training data into pretraining—can materially improve how well capabilities survive subsequent fine-tuning. Replay and dropout provide additional complementary gains, further suggesting that robustness can be influenced well before forgetting is observed downstream. More broadly, our results point to a promising direction for future work: building models that are easier to inherently easier adapt by designing the upstream training pipeline to make valuable capabilities more durable from the start.

2 Related Works

Catastrophic Forgetting A recurring challenge in sequential training is *catastrophic forgetting*: when a model is optimized on new data, its performance can deteriorate on behaviors it previously exhibited (McCloskey & Cohen, 1989). For language models, this phenomenon shows up in modern training pipelines. For example, instruction tuning and RLHF can trade off against preexisting capabilities, an effect often discussed as an "alignment tax" (Ouyang et al., 2022). Relatedly, several works show that behaviors introduced during safety finetuning can be quickly weakened or reversed by subsequent training on different objectives or data (Yang et al., 2023; Qi et al., 2023). These tradeoffs also appear in adjacent settings such as knowledge editing (Nishi et al., 2025) and unlearning (Maini et al., 2024). Beyond documenting the effect, recent work has started to map how training choices shape its severity: for instance, LoRA-style adaptation can alter forgetting dynamics (Biderman et al., 2024), and longer pretraining can change how brittle or persistent acquired capabilities are (Springer et al., 2025). In this paper, we focus on catastrophic forgetting of post-trained capabilities, and study what properties of an intermediate checkpoint determine whether capabilities persists under subsequent training.

Data Placement in Pretraining A line of recent works examine pre-training interventions for enforcing desired downstream capabilities and properties. Maini et al. (2025); O'Brien et al. (2025) propose filtering and augmenting data during pre-training to improve safety. Similarly, Sam et al. (2026) demonstrate that the impact of such interventions improves as they are introduced earlier in pre-training. While these works incorporate downstream tasks during pre-training, they extensively modify the pre-training corpus by incorporating data-augmentations and filtering of the dataset. Baek et al. (2026) demonstrate that mixing post-training data during pre-training can immediately improve in-domain performance relative to simply fine-tuning. In our work, we introduce an additional benefit of early exposure to post-training data: robustness to catastrophic forgetting during future training.

3 Preliminaries and Setting

Typically, a model developer (1) pretrains a model on a general web corpus and (2) post-trains the model on a target domain; downstream users then (3) fine-tune the model for their own purposes. Throughout this work, we'll describe stages (1) and (2) as upstream relative to stage (3) controlled by the downstream end user. Each stage is associated with a dataset: $\mathcal{D}_{\text{pre}}$ (general pretraining), $\mathcal{D}_{\text{post}}$ (post-training), and $\mathcal{D}_{\text{ft}}$ (fine-tuning). The post-training corpus is assumed to be much smaller than the pretraining web corpus, reflecting the practical regime where post-training data is relatively scarce.

We write $\mathcal{L}(\theta; \mathcal{D})$ for the loss of parameters θ evaluated on dataset $\mathcal{D}$. All losses are computed on held-out splits of the corresponding datasets.

Stage 1: Upstream Pretraining. The model is first pretrained on a general web corpus $\mathcal{D}_{\text{pre}}$, producing a pretrained checkpoint θ_{pre}. In some experiments, the upstream developer additionally mixes a fraction $\lambda \in [0, 1]$ of the post-training corpus $\mathcal{D}_{\text{post}}$ into this stage. Here, $\lambda = 0$ denotes no exposure to $\mathcal{D}_{\text{post}}$ during pretraining, while larger values of λ correspond to greater exposure to $\mathcal{D}_{\text{post}}$ during pretraining.

Stage 2: Upstream Post-training. Starting from θ_{pre}, the upstream developer post-trains the model on a relatively smaller corpus $\mathcal{D}_{\text{post}}$ to acquire a target capability or domain specialization, yielding the post-trained checkpoint θ_{post}. We measure performance on $\mathcal{D}_{\text{post}}$ immediately after this stage using the *immediate post-training loss*

$$\mathcal{L}_{\text{im}} := \mathcal{L}(\theta_{\text{post}}; \mathcal{D}_{\text{post}}).$$

Stage 3: Subsequent fine-tuning. A downstream user then fine-tunes the post-trained model on a new objective $\mathcal{D}_{\text{ft}}$, producing the checkpoint θ_{ft}. We measure performance on this new objective using the *downstream fine-tuning loss*

$$\mathcal{L}_{\text{ft}} := \mathcal{L}(\theta_{\text{ft}}; \mathcal{D}_{\text{ft}}).$$

Subsequent fine-tuning can degrade capabilities acquired during upstream post-training. To measure how much of the post-trained capability survives, we also evaluate θ_{ft} on $\mathcal{D}_{\text{post}}$, defining the *retained post-training loss*

$$\mathcal{L}_{\text{ret}} := \mathcal{L}(\theta_{\text{ft}}; \mathcal{D}_{\text{post}}).$$

Our central question is how upstream training choices affect downstream adaptation, retention of post-trained capabilities, and preservation of general-domain performance under subsequent fine-tuning.

3.1 Evaluation Methodology

Subsequent fine-tuning is inherently multi-objective. A downstream user may care not only about performance on the new fine-tuning objective $\mathcal{D}_{\text{ft}}$, but also about retaining capabilities acquired during upstream post-training on $\mathcal{D}_{\text{post}}$ and preserving more general capabilities associated with the pretraining distribution $\mathcal{D}_{\text{pre}}$. We therefore track three losses throughout this work: the downstream fine-tuning loss $\mathcal{L}_{\text{ft}}$, the retained post-training loss $\mathcal{L}_{\text{ret}}$, and the retained pretraining loss

$$\mathcal{L}_{\text{pre}} := \mathcal{L}(\theta_{\text{ft}}; \mathcal{D}_{\text{pre}}).$$

We use validation loss as our evaluation metric. Prior work has established loss as a reliable, scale-invariant proxy for capability: models with matched pretraining loss exhibit equivalent downstream task performance (Du et al., 2024; Gadre et al., 2024; Chen et al., 2025). This is particularly important at our training scales, where task accuracy is noisy and near random chance; in contrast, loss provides a continuous and smooth signal that enables fine-grained comparisons between models that may appear similar under coarse or discrete metrics.

Sweeping upstream training choices and hyperparameters yields checkpoints with different tradeoffs among these objectives. We summarize the best attainable tradeoffs using 2D *loss frontiers*: for each method, we

Pipeline	$\mathcal{D}_{\text{pre}}$	$\mathcal{D}_{\text{post}}$	$\mathcal{D}_{\text{ft}}$
Music → Chemistry	C4	MusicPile	ChemPile
Music → Instruction	C4	MusicPile	FLAN
Instruction → Chemistry	C4	FLAN	ChemPile
Instruction → Music	C4	FLAN	MusicPile

Table 1: Experimental instantiations of the three-stage pipeline. We vary the upstream post-training corpus $\mathcal{D}_{\text{post}}$ and downstream fine-tuning corpus $\mathcal{D}_{\text{ft}}$ while keeping the general pretraining corpus $\mathcal{D}_{\text{pre}}$ fixed to C4.

plot the Pareto-optimal set of checkpoints in a given 2D projection, i.e., those for which no other checkpoint from the same method achieves lower loss on both axes simultaneously. Our main analysis uses two complementary views: $(\mathcal{L}_{\text{ret}}, \mathcal{L}_{\text{ft}})$, which captures the tradeoff between retaining the post-trained capability and adapting to the downstream task, and $(\mathcal{L}_{\text{pre}}, \mathcal{L}_{\text{ret}})$, which captures the tradeoff between retaining broader pretraining capabilities and retaining the post-trained capability. Together, these views provide interpretable slices through the underlying three-objective tradeoff.

3.2 Experimental instantiations

Across experiments, we fix the general pretraining corpus to C4 and study the four three-stage instantiations shown in Table 1. These settings cover two qualitatively different forms of upstream capability acquisition, domain specialization and instruction tuning, and let us test whether the same robustness phenomena appear across different downstream fine-tuning objectives.

3.3 Model scales and training budgets

Unless otherwise stated, we use a SmolLM2-style architecture (Allal et al., 2025) at two scales: our primary experiments use a 135M-parameter model, and we additionally run a 1B-parameter variant to test whether the same qualitative patterns persist at larger scale. For the 135M experiments, we pretrain on approximately 10B tokens from $\mathcal{D}_{\text{pre}}$, optionally with additional exposure to $\mathcal{D}_{\text{post}}$ during Stage 1. Starting from the resulting checkpoint θ_{pre}, we perform Stage 2 post-training on $\mathcal{D}_{\text{post}}$ using AdamW with linear warmup and cosine decay. In all but the compute-matched experiments, training proceeds exclusively on $\mathcal{D}_{\text{post}}$, with no restriction on dataset repetitions: we apply early stopping and continue training as long as validation loss on $\mathcal{D}_{\text{post}}$ improves (up to a maximum budget of 2B tokens). This ensures that all models are trained to convergence on $\mathcal{D}_{\text{post}}$, and that differences in downstream retention are not attributable to unequal training duration. We then fine-tune each post-trained checkpoint θ_{post} on $\mathcal{D}_{\text{ft}}$ for a fixed token budget of 200M tokens with various learning rates.

Full optimizer settings, sweep ranges, and per-intervention details are provided in Appendix A.

4 Experiments and Results

We begin by asking whether pretraining-time mixing has any effect once post-training is run to convergence (Section 4.1). We then ask whether scarce post-training data should be mixed during pretraining or reserved for a dedicated post-training stage (section 4.2). Finally, we ask whether the benefits of mixing persist across broad hyperparameter sweeps and multiple pipeline instantiations (Section 4.3), before turning to replay and dropout as complementary post-training interventions (Section 4.4).

4.1 Immediate post-training performance does not reflect downstream retention

We begin with a controlled study of the mixing ratio λ, designed to isolate whether early exposure to $\mathcal{D}_{\text{post}}$ has any effect once upstream post-training is run to convergence using early stopping. Unlike the broad hyperparameter sweeps used in our main frontier analysis, these experiments fix the Stage 2 post-training procedure and vary only how much of $\mathcal{D}_{\text{post}}$ is seen during Stage 1 pretraining.

Setup. We fix the post-training configuration and vary only the pretraining mixing fraction $\lambda \in \{0, 0.25, 0.5, 0.75, 1.0\}$. We study three post-training dataset sizes $|\mathcal{D}_{\text{post}}| \in \{30\text{M}, 150\text{M}, 300\text{M}\}$ where $\mathcal{D}_{\text{post}} \subset$ MusicPile. Starting from each pretrained checkpoint, we post-train on $\mathcal{D}_{\text{post}}$ until convergence using a fixed hyperparameter configuration. To induce forgetting, we then fine-tune on $\mathcal{D}_{\text{ft}} \subset$ ChemPile, and report both the immediate post-training loss $\mathcal{L}_{\text{im}}$ and the retained post-training loss $\mathcal{L}_{\text{ret}}$ at a fixed Stage 3 learning rate of 5×10^{-5} (Figure 2a).

Result. Varying λ has little effect on immediate post-training performance: across dataset sizes, $\mathcal{L}_{\text{im}}$ remains nearly flat as the mixing fraction increases. In other words, once post-training is allowed to run to convergence, mixed and unmixed models reach similar performance on $\mathcal{D}_{\text{post}}$. However, these checkpoints behave very differently under subsequent fine-tuning. As λ increases, the retained post-training loss $\mathcal{L}_{\text{ret}}$ consistently decreases, indicating that models with more pretraining-time exposure to $\mathcal{D}_{\text{post}}$ forget less after subsequent downstream adaptation.

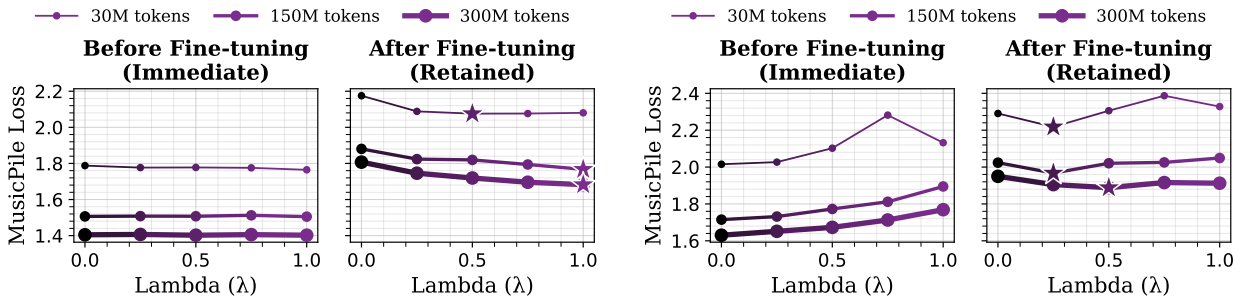
Takeaway. Pretraining-time mixing can substantially improve retention under subsequent fine-tuning even when it provides little or no benefit to immediate post-training performance.

4.2 Mixing and dedicated post-training play different roles under a fixed data budget

The experiment above in Section 4.1 shows that early exposure to post-training data can improve its downstream retention even when it has little effect on immediate post-training performance. However, those experiments do not isolate whether the benefit comes from *when* $\mathcal{D}_{\text{post}}$ is introduced or simply from the model seeing more total $\mathcal{D}_{\text{post}}$ tokens. We therefore ask a more controlled question: under a fixed $\mathcal{D}_{\text{post}}$ budget, should post-training data be mixed into pretraining at all, or is it better reserved for a dedicated post-training stage?

Setup. We fix the total number of $\mathcal{D}_{\text{post}}$ tokens seen across Stage 1 pretraining and Stage 2 post-training and vary only how that budget is allocated. For each mixing fraction $\lambda \in \{0, 0.25, 0.5, 0.75, 1.0\}$, we expose the model to a λ-fraction of $\mathcal{D}_{\text{post}}$ during pretraining and reserve the remaining $(1 - \lambda)$ fraction for post-training, so every model sees exactly one pass over $\mathcal{D}_{\text{post}}$ in total. Thus, $\lambda = 0$ assigns the full budget to dedicated post-training, while $\lambda = 1$ assigns it entirely to mixed pretraining. We evaluate this allocation study with $\mathcal{D}_{\text{post}} \subset$ MusicPile at three dataset sizes, $|\mathcal{D}_{\text{post}}| \in \{30\text{M}, 150\text{M}, 300\text{M}\}$, use $\mathcal{D}_{\text{ft}} \subset$ ChemPile for downstream fine-tuning, and report both $\mathcal{L}_{\text{im}}$ and $\mathcal{L}_{\text{ret}}$ after Stage 3 fine-tuning at a fixed learning rate of 5×10^{-5} (Figure 2b).

Result. Figure 2b reveals a clear tradeoff. As λ increases, the immediate post-training loss $\mathcal{L}_{\text{im}}$ worsens: under a fixed $\mathcal{D}_{\text{post}}$ budget, allocating more data to pretraining leaves fewer tokens for the post-training stage that exclusively optimizes for $\mathcal{D}_{\text{post}}$. The downstream picture is different. Increasing λ consistently improves retained performance after Stage 3 fine-tuning, lowering $\mathcal{L}_{\text{ret}}$ across dataset sizes. As a result, the best immediate post-training performance is achieved by allocating all of $\mathcal{D}_{\text{post}}$ to Stage 2, while the



(a) Varying mixture fraction λ.

(b) Compute-matched setting.

Figure 2: Left: As the mixture fraction λ increases, immediate MusicPile loss after post-training remains nearly constant, while retained MusicPile loss after downstream fine-tuning on ChemPile improves. This shows that the benefits of mixing can be *latent*: they may not be visible immediately after post-training, but emerge after subsequent fine-tuning. **Right:** In a compute-matched setting where total MusicPile exposure is held fixed across pretraining and post-training, increasing λ worsens immediate MusicPile loss after post-training but improves retained MusicPile loss after downstream fine-tuning. Thus, even under a fixed MusicPile token budget, allocating some exposure earlier in training yields better retention.

best retained performance after downstream adaptation is achieved at a nonzero mixture fraction. Even under a fixed data budget, the optimum therefore lies between the two extremes of all-post-training and all-pretraining allocation. This suggests that dedicated post-training and pretraining-time mixing are doing something meaningfully different: concentrating $\mathcal{D}_{\text{post}}$ in Stage 2 yields stronger immediate fitting to the post-training domain, while earlier exposure makes that capability less brittle under later adaptation. We provide a theoretical understanding of this in Section 5.

Takeaway: Under a fixed $\mathcal{D}_{\text{post}}$ budget, the best immediate post-training performance occurs when all data is reserved for Stage 2, but the best retained performance after downstream fine-tuning occurs at a positive mixture fraction.

4.3 Mixed pretraining improves the loss frontier across hyperparameter sweeps

The controlled studies above isolate two key phenomena: the benefit of mixing can be latent, and under a fixed post-training-data budget the best retained performance is achieved at a nonzero mixture fraction. We now ask whether these conclusions persist once we move beyond controlled comparisons to the more realistic setting in which both post-training and downstream fine-tuning offer many tunable degrees of freedom. In practice, an upstream developer can vary post-training hyperparameters to reach different tradeoffs between adaptation and retention, while a downstream end user may likewise vary fine-tuning hyperparameters to target different operating points. We therefore evaluate each upstream strategy not by a single checkpoint, but by the frontier of attainable checkpoints it induces across broad Stage 2 post-training sweeps and Stage 3 fine-tuning sweeps.

Setup. For each pipeline in Table 1, we sweep Stage 2 post-training hyperparameters under both unmixed and mixed pretraining, then fine-tune every resulting checkpoint on the downstream objective using a range of Stage 3 learning rates, and evaluate the paired frontier views shown in Figure 3. Within each pipeline, the left panel plots retained post-training loss against downstream fine-tuning loss, while the right panel plots retained pretraining loss against retained post-training loss.

Result. Across all four pipelines, mixed pretraining consistently shifts the frontier relative to unmixed pretraining. In the $(\mathcal{L}_{\text{ret}}, \mathcal{L}_{\text{ft}})$ view, mixing yields lower retained post-training loss at matched downstream fine-tuning loss, indicating greater robustness of the post-trained capability under subsequent adaptation. In the $(\mathcal{L}_{\text{pre}}, \mathcal{L}_{\text{ret}})$ view, mixing also improves the tradeoff between preserving broader pretraining capabilities

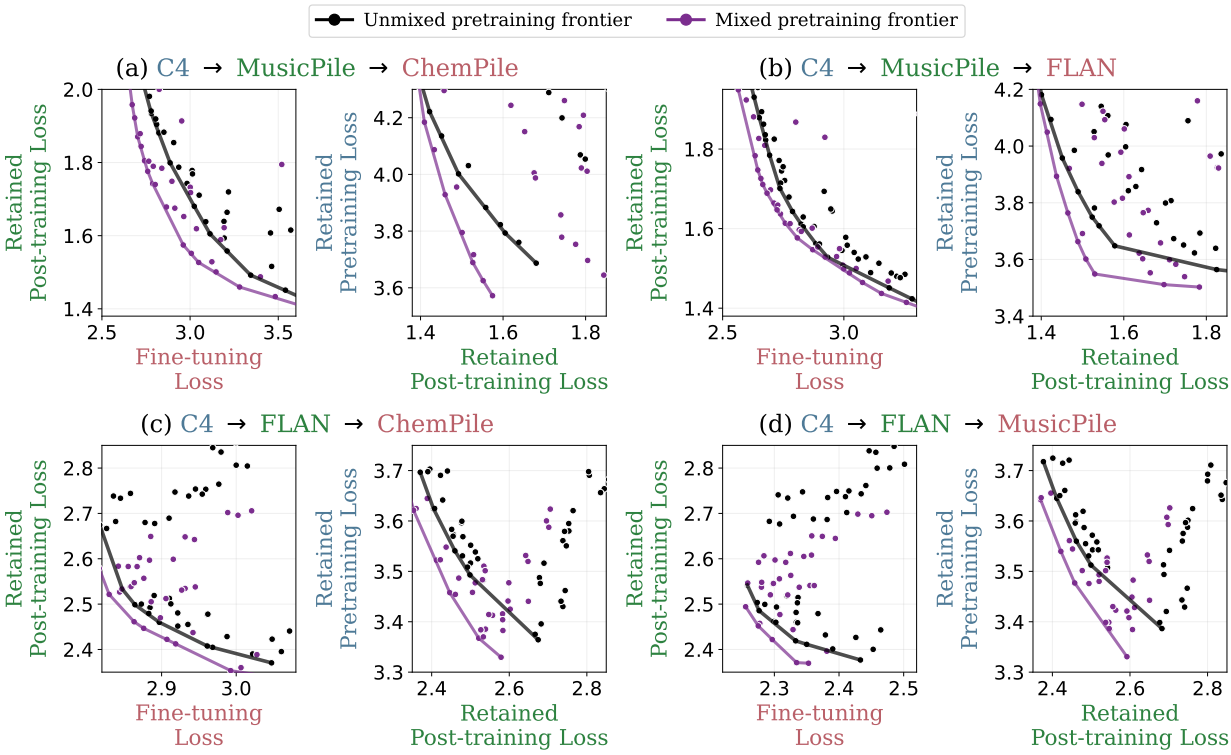


Figure 3: Mixing during pretraining improves the tradeoff frontier across four training pipelines (135M). Each panel corresponds to one 3-stage pipeline. Within each panel, the left plot shows retained post-training loss versus downstream fine-tuning loss, and the right plot shows retained pretraining loss versus retained post-training loss. **Black** denotes the frontier obtained from unmixed pretraining, and **purple** denotes the frontier obtained from mixed pretraining. Across all four pipelines, mixing shifts the frontier toward lower retained post-training loss, lower retained pretraining loss, and lower downstream fine-tuning loss, indicating that early exposure to post-training data can improve its downstream retention after subsequent fine-tuning.

and preserving the post-trained capability. These gains appear across both domain and behavioral post-training settings, suggesting that the benefit of mixing is not confined to a single dataset pair or narrow training regime. We further verify that the same qualitative frontier improvement appears in our 1B experiments, suggesting that the benefit of mixing persists beyond the small-model setting (Figures 8 and 9).

Takeaway: Across hyperparameter sweeps and training pipelines, mixed pretraining consistently improves the attainable tradeoffs among downstream fine-tuning performance, retained post-training performance, and retained pretraining performance.

4.4 Replay and dropout provide complementary gains

Pretraining-time mixing is one way to shape how the model acquires the target post-training capability, but it is not the only one. We next study two alternative upstream interventions that act during Stage 2 post-training itself: *replay*, which mixes a small amount of general-domain data into post-training, and *dropout*, which regularizes the post-training update. We view both as interventions on *how* the model learns $\mathcal{D}_{\text{post}}$, rather than simply how much performance it achieves immediately after post-training.

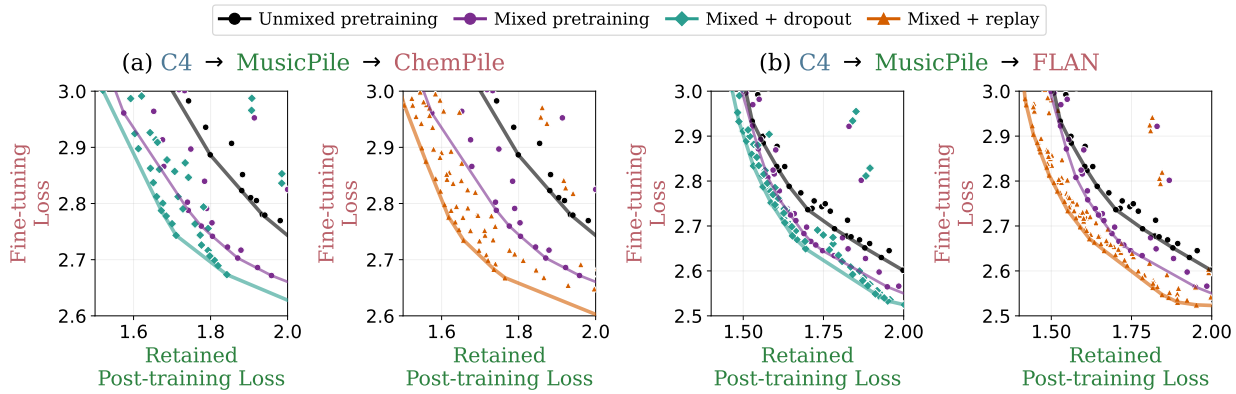


Figure 4: Replay and dropout provide complementary gains on top of mixed pretraining. Each subfigure shows one 3-stage pipeline. Within each subfigure, the **left** panel compares unmixed pretraining, mixed pretraining, and mixed pretraining + dropout, while the **right** panel compares unmixed pretraining, mixed pretraining, and mixed pretraining + replay. Across both downstream settings, adding dropout or replay to mixed pretraining further shifts the fine-tuning–retention frontier, indicating that these post-training interventions provide complementary gains rather than replacing the effect of pretraining-time mixing.

Setup. We evaluate replay and dropout in the same three-stage framework as above, using broad Stage 2 hyperparameter sweeps and, for every resulting checkpoint, sweeping Stage 3 fine-tuning learning rates before measuring the resulting frontier between downstream fine-tuning loss $\mathcal{L}_{\text{ft}}$ and retained post-training loss $\mathcal{L}_{\text{ret}}$. Replay is implemented by mixing a small fraction (1% following Bethune et al. (2025)) of general-domain data from $\mathcal{D}_{\text{pre}}$ into post-training, while dropout is applied during post-training as a regularizer.

Result. Figure 4 shows that both replay and dropout further improve the fine-tuning–retention frontier relative to mixed pretraining alone. Replay encourages the model to acquire the post-training domain without simply overwriting broader pretraining features, while dropout may promote more distributed and robust representations during Stage 2 learning. These gains persist after downstream fine-tuning in both representative pipelines we study, indicating that post-training interventions can meaningfully improve robustness to later adaptation.

At 1B, the effect of replay on the $(\mathcal{L}_{\text{ft}}, \mathcal{L}_{\text{ret}})$ frontier is weaker. However, replay remains useful for preserving broader general-domain performance, suggesting that its benefits may shift across scales and objectives (Figures ?? and ??).

Complementarity with mixing. Neither replay nor dropout eliminates the value of pretraining-time mixing. Instead, both remain complementary to mixing: the strongest frontiers are obtained by combining post-training interventions with mixed pretraining. This reinforces the broader picture developed above. Pretraining-time mixing plays a distinct role by changing when the model first encounters the post-training capability, while replay and dropout shape how that capability is learned during Stage 2.

Takeaway: Replay and dropout are simple upstream alternatives to pretraining-time mixing that can also improve robustness to subsequent fine-tuning, and their gains are complementary with mixing.

5 Theoretical Analysis of Early Exposure

We find that mixing even a small fraction of $\mathcal{D}_{\text{post}}$ into pretraining substantially improves retention under subsequent fine-tuning, and that this effect is complementary to various post-training regularizations applied during post-training. This is striking, as there is little reason to expect that a small change to the pretraining corpus should persistently shape retention two stages downstream. Moreover, these benefits persist across a range of post-training interventions, suggesting that mixing operates on a different axis than techniques that regularize the post-training or fine-tuning stages. Broadly, our findings indicate that early exposure has a unique effect on how capabilities are implemented in the model, which then propagates through subsequent training stages.

We now formally characterize how early exposure affects the way post-trained capabilities are implemented in the model and their vulnerability to future forgetting. We analyze our three-stage pipeline in a two layer linear model, which enables a precise characterization of feature learning during pre-training and its impact on subsequent training dynamics.

Setup. We train a two-layer linear network $\theta = \boldsymbol{W}_1\boldsymbol{W}_2$ sequentially on three regression tasks simulating $\mathcal{D}_{\text{pre}}$, $\mathcal{D}_{\text{post}}$, and $\mathcal{D}_{\text{ft}}$, using gradient descent on the squared loss $\mathcal{L}(\theta;\mathcal{D}) = \boldsymbol{E}[||\theta\boldsymbol{x} - \boldsymbol{y}||^2]$. Each task is defined by an input distribution and a ground-truth linear map $\boldsymbol{A}^t$, with $\boldsymbol{y} = \boldsymbol{A}^t\boldsymbol{x}$. Following Springer et al. (2025), we assume all tasks share singular vectors $\boldsymbol{U}, \boldsymbol{V}$ so that $\boldsymbol{A}^t = \boldsymbol{U}\boldsymbol{\Sigma}_t\boldsymbol{V}^\top$; the singular values of $\boldsymbol{\Sigma}_t$ are the *features* of task t.

Feature structure. We consider the input-space to be partitioned into three blocks, exhibiting different behaviors across the tasks:

- *Invariant features* ($n-2k$ features) have identical singular values across $\boldsymbol{A}^{\text{gen}}$, $\boldsymbol{A}^{\text{spec}}$, and $\boldsymbol{A}^{\text{ft}}$.
- *Inconsistent features* (k features) have shared dimensions where the tasks disagree—the singular values differ between $\boldsymbol{A}^{\text{gen}}$, $\boldsymbol{A}^{\text{spec}}$, and $\boldsymbol{A}^{\text{ft}}$.
- *Specialized features* (k features) are active only on $\mathcal{D}_{\text{post}}$, while having zero covariance under pretraining and downstream distributions.

Task definitions. The pretraining task $\mathcal{D}_{\text{pre}}$ draws inputs from $x \sim \mathcal{N}(0, \boldsymbol{I}_{n-k})$, activating only shared dimensions. The post-training task $\mathcal{D}_{\text{post}}$ draws inputs from $x \sim \mathcal{N}(0, \boldsymbol{I}_n)$, activating all dimensions including the specialized ones. Like pretraining, the downstream task $\mathcal{D}_{\text{ft}}$ draws inputs from $x \sim \mathcal{N}(0, \boldsymbol{I}_{n-k})$—critically, it does not activate the specialized features. We model *early exposure* by training on the distribution $\mathcal{D}_{\text{mixed}} = (1-\alpha)\mathcal{D}_{\text{gen}} + \alpha\mathcal{D}_{\text{spec}}$. Full details and formal assumptions are given in Appendix C.

5.1 Early Exposure Learns Different Features

We first characterize what features each pretraining strategy learns.

Theorem 5.1 (*Informal:* Only Mixing Learns Specialized Features)**.** *Let θ^{mixed} and $\theta^{unmixed}$ be the parameters learned by pretraining on $\mathcal{D}_{mixed}$ and $\mathcal{D}_{gen}$, respectively, after sufficient training. Then θ^{mixed} learns the specialized features, while $\theta^{unmixed}$ does not.*

The key mechanism is that linear networks learn features in descending order of their singular value (Gidel et al., 2019; Springer et al., 2025), with remaining features staying near zero. Without early exposure, the k specialized features have the lowest singular values and are not learned. Early exposure, on the other hand, boosts the effective singular value above the learning threshold. See Appendix C for the formal statement.

Impact of a Small Mixing Ratio The mixing fraction α need not be large to have an impact. Suppose the specialized feature has singular value $\beta > 0$ on $\mathcal{D}_{\text{post}}$. Without mixing, specialized features have zero effective singular value and are never learned. Mixing an α-fraction of $\mathcal{D}_{\text{post}}$ boosts the effective singular value to $\alpha\beta$, so when β is sufficiently large, even small α suffices to cross the threshold at which specialized features are learned instead. Empirically, we also observe material benefits for early exposure even when the quantity of mixed data is small relative to the total pre-training corpus.

5.2 Post-Training Primarily Reuses Existing Features

Next, we study the impact of the different features learned by early exposure on the post-training process.

Theorem 5.2 (*Informal*: Post-training on θ^{mixed} versus θ^{unmixed})**.** *After sufficient post-training on $\mathcal{D}_{post}$, θ^{mixed}_{post} converges to parameters that leverage specialized features to reduce post-training loss, while $\theta^{unmixed}_{post}$ converges to parameters that only modify the inconsistent features.*

The key insight is that features absent at initialization remain absent throughout post-training: if $\Sigma_{ii} = 0$ at the start of post-training, it stays 0. Since θ^{unmixed} never learned the specialized features, post-training from this checkpoint can only reduce loss on $\mathcal{D}_{\text{post}}$ by distorting the inconsistent features. Post-training from θ^{mixed}, having learned the specialized features during pretraining, can additionally improve loss on $\mathcal{D}_{\text{post}}$ using the specialized features. We will next show that these

Cost of Specialization to General Performance. Our analysis shows that specialization in the absence of early exposure is achieved by modifying inconsistent features. This suggests that such specialization must come at a cost to general language modeling capability. The specialized features learned during early exposure, on the other hand, provide a mechanism for specialization that does not impact other tasks. Our empirical results are consistent with this: post-training after early exposure suffers less degradation of C4 loss than post-training a base model without early exposure, as observed in Figure 3.

5.3 Characterizing Downstream Forgetting Behavior

Finally, we show that the different feature usage established above directly determines forgetting under downstream fine-tuning.

Theorem 5.3 (*Informal:* $\theta^{\text{unmixed}}_{\text{post}}$ experiences more forgetting than $\theta^{\text{mixed}}_{\text{post}}$)**.** *Let $\Delta_{mixed}, \Delta_{unmixed}$ denote the increase in loss on $\mathcal{D}_{post}$ after fine-tuning on $\mathcal{D}_{ft}$, for the mixed and unmixed checkpoints, respectively. Then $\Delta_{unmixed} \geq \Delta_{mixed}$.*

Because $\mathcal{D}_{\text{ft}}$ does not activate the specialized dimensions, gradient updates during fine-tuning have zero projection along those directions. Inconsistent features, however, overlap with $\mathcal{D}_{\text{ft}}$ and are overwritten. Without early exposure, all the loss reduction on $\mathcal{D}_{\text{post}}$ is implemented in the inconsistent features and is therefore vulnerable to erasure during fine-tuning. With early exposure, loss reduction is also implemented in protected specialized features, enabling it to persist after further training. This provides a formal account of the frontier shift observed empirically: the mixed checkpoint's retention advantage traces back to feature learning during pretraining.

Summary Ultimately, our analysis shows that even limited early exposure to post-training data during pre-training induces the formation of *specialized features* for that domain. These features remain decoupled from both $\mathcal{D}_{\text{gen}}$ and $\mathcal{D}_{\text{post}}$, rendering them resistant to overwriting during fine-tuning—fine-tuning gradients have negligible projection along these directions. In contrast, without early exposure, post-training relies on modifying broadly shared features to reduce loss, making these adaptations inherently fragile and prone to forgetting under subsequent fine-tuning.

6 Conclusion

Main themes. This thesis advances three connected claims. First, how well a capability survives later fine-tuning cannot be read off from how well a model performs on that capability immediately after it is acquired; two training recipes that look equivalent at the handoff can diverge substantially once the model is adapted downstream. Second, the *manner* in which a capability is learned — when it is introduced during training, how it is presented, and what else the model is learning at the same time — shapes how durable that capability will be under later updates. Third, upstream training offers not a single knob but a family of interventions: early exposure during pretraining, replay during post-training, and regularization during post-training each shift the retention–adaptation frontier, and their effects are largely complementary rather than substitutable. The upstream design space for durability is richer than any single technique. Together, these observations reframe robustness to fine-tuning from a downstream problem to be mitigated into a design objective of upstream training.

Limitations. The present study leaves several important questions open. We examine a controlled setting in which upstream and downstream training data are drawn from distinct domains; it remains unclear how the benefits of early exposure behave when the two distributions are closely related, or when they actively conflict. Our experiments also restrict the amount of post-training data introduced during pretraining to at most a single pass over that corpus; we do not characterize what happens when the post-training data is repeated much more aggressively — for instance, allocating several percent of the total pretraining budget to post-training data, which would imply many repetitions of a small corpus. Our experiments span two model scales and a fixed pretraining token budget, so we cannot directly speak to whether the same upstream interventions continue to pay off at production-scale training. We focus on a single downstream adaptation method and do not characterize how early exposure interacts with parameter-efficient or preference-based fine-tuning schemes. Finally, our theoretical account is developed in a simplified model and is best read as a mechanistic illustration rather than a proof about modern language models.

Future work. Several directions follow naturally. A fuller characterization of when early exposure helps — across varying degrees of overlap between upstream and downstream data, across a wider range of exposure levels, and into regimes of heavy repetition of post-training data — would sharpen practical guidance for upstream developers. Extending the study to larger models and longer training runs would test whether upstream interventions continue to shift the retention frontier at scale. A complementary line of work is algorithmic: whether new objectives or regularizers applied later in training can approximate the representational effect of early exposure, without modifying the pretraining corpus. More broadly, our findings suggest that the goal of upstream training should not only be to teach models useful capabilities, but to teach them in forms that remain stable under the updates they are likely to encounter later — making durability, rather than immediate performance, a central target of model development.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. Smollm2: When smol goes big – data-centric training of a small language model, 2025. URL https://arxiv.org/abs/2502.02737.

Christina Baek, Ricardo Pio Monti, David Schwab, Amro Abbas, Rishabh Adiga, Cody Blakeney, Maximilian Böther, Paul Burstein, Aldo Gael Carranza, Alvin Deng, Parth Doshi, Vineeth Dorna, Alex Fang, Tony Jiang, Siddharth Joshi, Brett W. Larsen, Jason Chan Lee, Katherine L. Mentzer, Luke Merrick, Haakon Mongstad, Fan Pan, Anshuman Suri, Darren Teh, Jason Telanoff, Jack Urbanek, Zhengping Wang, Josh Wills, Haoli Yin, Aditi Raghunathan, J. Zico Kolter, Bogdan Gaza, Ari Morcos, Matthew Leavitt, and Pratyush Maini. The finetuner's fallacy: When to pretrain with your finetuning data, 2026. URL https://arxiv.org/abs/2603.16177.

Louis Bethune, David Grangier, Dan Busbridge, Eleonora Gualdoni, Marco Cuturi, and Pierre Ablin. Scaling laws for forgetting during finetuning with pretraining data injection, 2025. URL https://arxiv.org/abs/2502.06042.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John P. Cunningham. Lora learns less and forgets less, 2024. URL https://arxiv.org/abs/2405.09673.

Yangyi Chen, Binxuan Huang, Yifan Gao, Zhengyang Wang, Jingfeng Yang, and Heng Ji. Scaling laws for predicting downstream performance in llms, 2025. URL https://arxiv.org/abs/2410.08527.

Zhengxiao Du, Aohan Zeng, Yuxiao Dong, and Jie Tang. Understanding emergent abilities of language models from the loss perspective. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Samir Yitzhak Gadre, Georgios Smyrnis, Vaishaal Shankar, Suchin Gururangan, Mitchell Wortsman, Rulin Shao, Jean Mercat, Alex Fang, Jeffrey Li, Sedrick Keh, Rui Xin, Marianna Nezhurina, Igor Vasiljevic, Jenia Jitsev, Luca Soldaini, Alexandros G. Dimakis, Gabriel Ilharco, Pang Wei Koh, Shuran Song, Thomas Kollar, Yair Carmon, Achal Dave, Reinhard Heckel, Niklas Muennighoff, and Ludwig Schmidt. Language models scale reliably with over-training and on downstream tasks, 2024. URL https://arxiv.org/abs/2403.08540.

Gauthier Gidel, Francis Bach, and Simon Lacoste-Julien. Implicit regularization of discrete gradient dynamics in linear neural networks, 2019. URL https://arxiv.org/abs/1904.13262.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, March 2017. ISSN 1091-6490. doi: 10.1073/pnas.1611835114. URL http://dx.doi.org/10.1073/pnas.1611835114.

Suhas Kotha and Percy Liang. Replaying pre-training data improves fine-tuning, 2026. URL https://arxiv.org/abs/2603.04964.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C. Lipton, and J. Zico Kolter. Tofu: A task of fictitious unlearning for llms, 2024. URL https://arxiv.org/abs/2401.06121.

Pratyush Maini, Sachin Goyal, Dylan Sam, Alex Robey, Yash Savani, Yiding Jiang, Andy Zou, Matt Fredrikson, Zacharcy C. Lipton, and J. Zico Kolter. Safety pretraining: Toward the next generation of safe ai, 2025. URL https://arxiv.org/abs/2504.16980.

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

Kento Nishi, Rahul Ramesh, Maya Okawa, Mikail Khona, Hidenori Tanaka, and Ekdeep Singh Lubana. Representation shattering in transformers: A synthetic study with knowledge editing, 2025. URL https://arxiv.org/abs/2410.17194.

Kyle O'Brien, Stephen Casper, Quentin Anthony, Tomek Korbak, Robert Kirk, Xander Davies, Ishan Mishra, Geoffrey Irving, Yarin Gal, and Stella Biderman. Deep ignorance: Filtering pretraining data builds tamper-resistant safeguards into open-weight llms, 2025. URL https://arxiv.org/abs/2508.06601.

Team Olmo, :, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, Pradeep Dasigi, Robert Berry, Saumya Malik, Saurabh Shah, Scott Geng, Shane Arora, Shashank Gupta, Taira Anderson, Teng Xiao, Tyler Murray, Tyler Romero, Victoria Graf, Akari Asai, Akshita Bhagia, Alexander Wettig, Alisa Liu, Aman Rangapur, Chloe Anastasiades, Costa Huang, Dustin Schwenk, Harsh Trivedi, Ian Magnusson, Jaron Lochner, Jiacheng Liu, Lester James V. Miranda, Maarten Sap, Malia Morgan, Michael Schmitz, Michal Guerquin, Michael Wilson, Regan Huff, Ronan Le Bras, Rui Xin, Rulin Shao, Sam Skjonsberg, Shannon Zejiang Shen, Shuyue Stella Li, Tucker Wilde, Valentina Pyatkin, Will Merrill, Yapei Chang, Yuling Gu, Zhiyuan Zeng, Ashish Sabharwal, Luke Zettlemoyer, Pang Wei Koh, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. Olmo 3, 2025. URL https://arxiv.org/abs/2512.13961.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. 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 Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

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!, 2023. URL https://arxiv.org/abs/2310.03693.

Dylan Sam, Sachin Goyal, Pratyush Maini, Alexander Robey, and J. Zico Kolter. When should we introduce safety interventions during pretraining?, 2026. URL https://arxiv.org/abs/2601.07087.

Jacob Mitchell Springer, Sachin Goyal, Kaiyue Wen, Tanishq Kumar, Xiang Yue, Sadhika Malladi, Graham Neubig, and Aditi Raghunathan. Overtrained language models are harder to fine-tune, 2025. URL https://arxiv.org/abs/2503.19206.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time, 2022a. URL https://arxiv.org/abs/2203.05482.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo-Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. Robust fine-tuning of zero-shot models, 2022b. URL https://arxiv.org/abs/2109.01903.

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, 2025. URL https://arxiv.org/abs/2505.09388.

Xianjun Yang, Xiao Wang, Qi Zhang, Linda Petzold, William Yang Wang, Xun Zhao, and Dahua Lin. Shadow alignment: The ease of subverting safely-aligned language models, 2023. URL https://arxiv.org/abs/2310.02949.

Contributions and Acknowledgments

Lawrence Feng led the project and conducted all the main experiments.

Gaurav R. Ghosal, Ziqian Zhong, Jacob Mitchell Springer, and Aditi Raghunathan were involved throughout the project. I (Lawrence) want to thank everyone for their invaluable guidance, mentorship, and support.

We'd also like to thank Christina Baek for her insights on the dynamics of pretraining and fine-tuning, and Kevin Li for his feedback on earlier versions of this work.

We'd also like to thank the Babel cluster, the Babel GPUs, and the Babel admins, without whom this project could not have happened.

A Training Details

A.1 SmolLM2-1B Model Architecture

Table 2: SmolLM2-1B model architecture (custom config interpolated from SmolLM2 family).

Parameter	Value
Parameters	1.03B
Hidden dimension	1,728
Attention heads	27
Layers	24
Head dimension	64
Query groups	27 (MHA)
MLP intermediate size	4,608
Vocabulary size	49,152
Context length	8,192 (max), 1,024 (training)
Normalization	RMSNorm
Position encoding	RoPE (base=100,000)

A.2 Dataset Statistics

Table 3: 135M Parameter Experiments

Dataset	Split	Tokens	Description
C4	Train	8.7B	General web text pretraining corpus
MusicPile	Train	0.3B	Music-domain text corpus
ChemPile	Train	0.3B	Chemistry-domain text corpus
FLAN	Train	0.3B	Instruction-tuning dataset

Table 4: 1B Parameter Experiments

Dataset	Split	Tokens	Description
C4	Train	19.7B	General web text pretraining corpus
MusicPile	Train	0.3B	Music-domain text corpus
ChemPile	Train	0.3B	Chemistry-domain text corpus
FLAN	Train	0.3B	Instruction-tuning dataset

A.3 Optimizer Configuration

Table 5: Optimizer configuration used across all experiments.

Parameter	Value
Optimizer	AdamW
β_1	0.9
β_2	0.95
Gradient clipping	1.0 (max norm)
Precision	bf16-mixed

A.4 1B Stage 1 Pretraining

Table 6: Stage 1 pretraining configuration for 1B experiments.

Parameter	Value	Notes
Total tokens	20B	Chinchilla-optimal for 1.03B params
C4 corpus	21.0B tokens	120 shards, tokenized
Learning rate	5e-4	Peak LR
Minimum LR	5e-5	Cosine decay target
Warmup steps	1,000	
Global batch size	512	
Micro batch size	30	Per-GPU
Eval interval	1,000 steps	
Save interval	1,000 steps	
GPUs	8× L40S	Single node
Seed	42	

A.5 Post-training Hyperparameters

Table 7: FFT hyperparameter search space for Stage 2 post-training (135M).

Parameter	Values	Notes
Learning rate	{1e-4, 2e-4, 5e-4, 1e-3, 5e-3}	Peak LR
Minimum LR	5e-5	Cosine decay target
Dropout	{0.0, 0.02, 0.05}	embed/attn/resid/mlp
Weight decay	0.1	Fixed
Warmup steps	500	Fixed
Batch size	{192, 480, 896}	Global batch size
Max tokens	2B	With early stopping

Table 8: Stage 2 FFT hyperparameter search space for 1B-scale post-training.

Parameter	Values	Notes
Learning rate	{1e-5, 2e-5, 5e-5, 1e-4, 2e-4}	Peak LR
Minimum LR	5e-5	Cosine decay target
Dropout	0.0	Fixed (see §A.7 for ablation)
Weight decay	0.1	Fixed
Warmup steps	100	
Global batch size	512	
Micro batch size	30	Per-GPU
Max tokens	2B	With early stopping
CPT budget	300M	Tokens per epoch (subsampled)
Early stopping patience	3	Evaluation intervals
Evaluation interval	100 steps	
GPUs	8× L40S	Single node
Seed	40	

A.6 LoRA Configuration

Table 9: LoRA hyperparameter configuration for Stage 2 post-training.

Parameter	Values	Notes
LoRA rank (r)	64	Fixed
LoRA alpha (α)	128	Fixed, $\alpha/r = 2$
LoRA dropout	{0.0, 0.02, 0.05}	Same as FFT dropout
LoRA targets	projection, mlp, head	Q/K/V excluded
Learning rate	{1e-4, 2e-4, 5e-4, 1e-3, 5e-3}	Same as FFT
Weight decay	0.1	Same as FFT
Other parameters	Same as FFT (Table 7)	

A.7 1B Stage 2 CPT: Dropout Ablation

Table 10: Dropout ablation configuration for 1B Stage 2 CPT. MusicPile CPT pipeline only. 12 runs total (2 $\lambda \times$ 3 LRs $\times$ 2 dropout rates).

Parameter	Values	Notes
Dropout	{0.02, 0.05}	embed/attn/resid/mlp
Learning rate	{1e-5, 2e-5, 5e-5}	Best 3 from baseline
λ	{0.0, 1.0}	MusicPile mixing only
CPT dataset	MusicPile	Priority pipeline
Other parameters	Same as baseline (Table 8)	

B Additional Plots

B.1 135M Dropout and Replay Frontiers with Retained Pretraining Loss (C4)

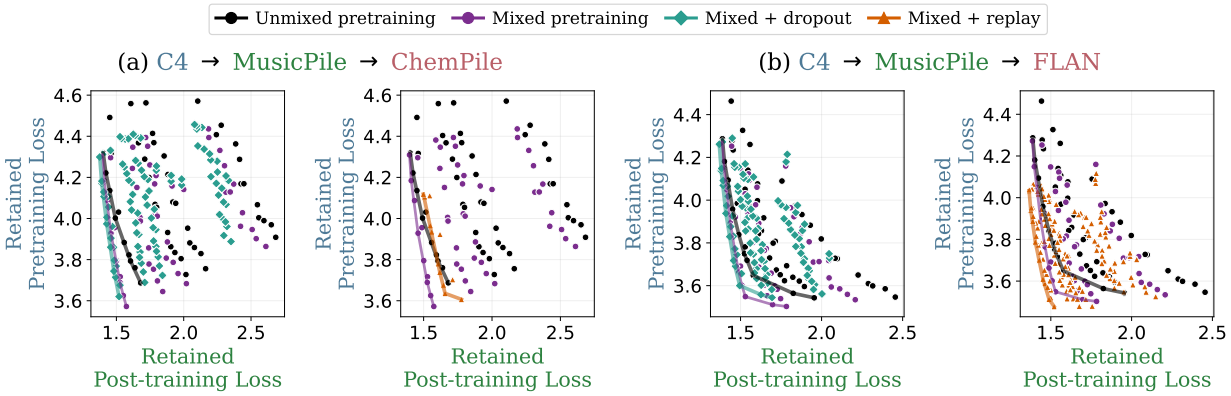


Figure 5: Dropout and replay preserve broader pretraining capability in addition to the post-training capability (135M). Companion to Figure 4, plotting the same Stage 2 hyperparameter sweeps against retained pretraining loss on C4 instead of downstream fine-tuning loss. Within each pipeline, the left panel compares unmixed pretraining, mixed pretraining, and mixed pretraining + dropout, and the right panel compares unmixed pretraining, mixed pretraining, and mixed pretraining + replay. Across both downstream settings, adding dropout or replay to mixed pretraining further lowers retained pretraining loss at matched retained post-training loss, indicating that these post-training interventions protect broader pretraining capabilities as well as the targeted post-trained capability. **(a)** C4 → MusicPile → ChemPile. **(b)** C4 → MusicPile → FLAN.

B.2 Additional 135M Dropout and Replay Frontiers (without mixing)

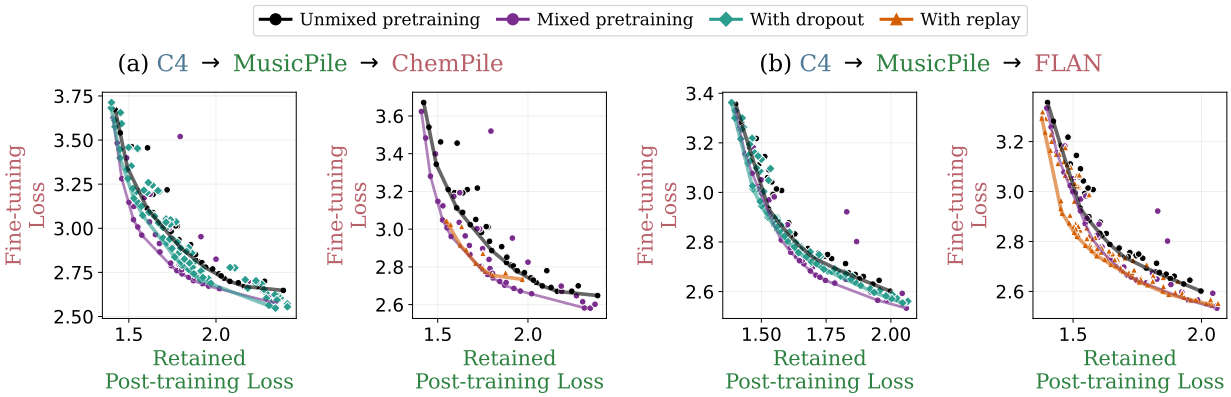


Figure 6: Dropout and replay applied without pretraining-time mixing (135M). To isolate the effect of post-training interventions from pretraining-time mixing, each panel applies dropout or replay on top of *unmixed* pretraining ($\lambda = 0$), with the mixed-pretraining frontier shown for reference. Within each pipeline, the left panel adds dropout during Stage 2 post-training; the right panel adds a small fraction (1%) of general-domain replay. Both interventions shift the fine-tuning–retention frontier, but less than pretraining-time mixing alone, reinforcing that mixing acts on a distinct axis from Stage 2 regularization.

B.3 135M LoRA Experiments

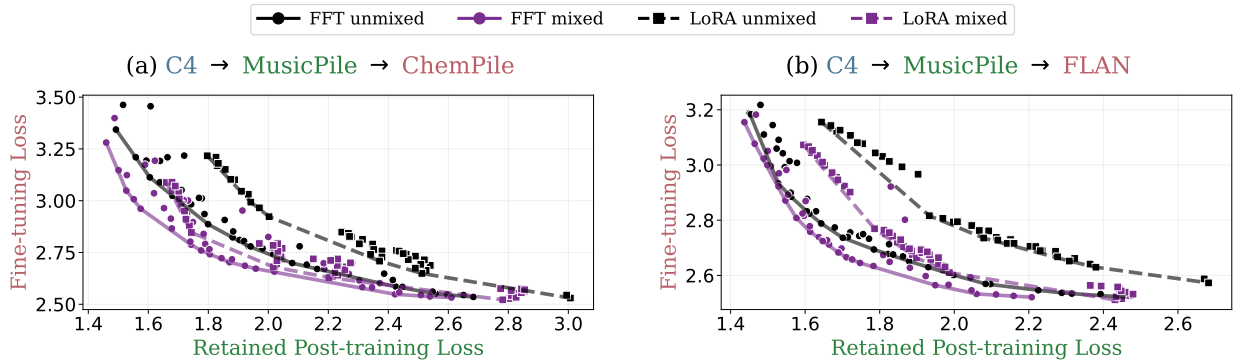


Figure 7: FFT vs LoRA fine-tuning–retention frontiers (135M). Each panel shows four frontiers obtained by sweeping Stage 2 post-training hyperparameters and Stage 3 fine-tuning learning rates: FFT with unmixed pretraining (black circles, solid), FFT with mixed pretraining (purple circles, solid), LoRA with unmixed pretraining (black squares, dashed), and LoRA with mixed pretraining (purple squares, dashed). Mixed pretraining improves both the FFT and LoRA frontiers in both downstream settings, and FFT generally attains a better fine-tuning–retention tradeoff than LoRA at matched upstream configurations. This suggests that the benefit of pretraining-time mixing is not specific to a particular fine-tuning method.

B.4 1B Experiments

Black denotes the frontier obtained from unmixed pretraining, and **purple** denotes the frontier obtained from mixed pretraining.

B.4.1 Mixing Frontiers

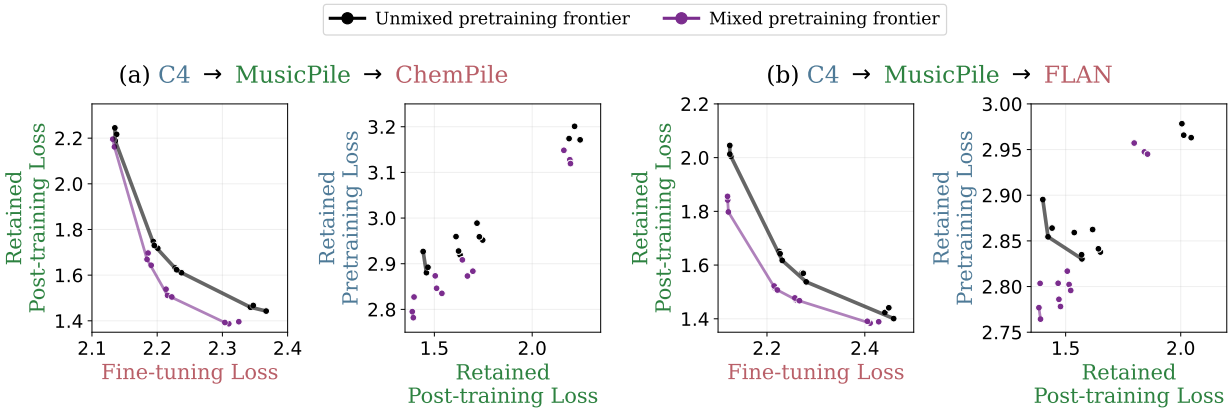


Figure 8: Mixing frontiers at 1B, MusicPile post-training pipelines. Companion to Figure 3 at 1B scale. Within each pipeline, the left panel plots retained post-training loss against downstream fine-tuning loss, and the right panel plots retained pretraining loss against retained post-training loss. As at the 135M scale, mixed pretraining consistently shifts the frontier toward lower retained post-training loss, lower retained pretraining loss, and lower downstream fine-tuning loss, indicating that the benefit of early exposure to post-training data persists beyond the small-model setting.

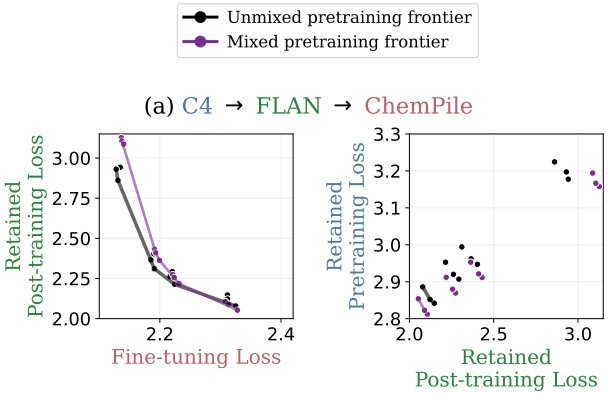


Figure 9: Mixing frontiers at 1B, FLAN post-training pipeline. Left: retained post-training loss vs fine-tuning loss. Right: retained pretraining loss (C4) vs retained post-training loss. Mixed pretraining likewise improves the frontier when the post-training corpus is instruction-tuning data rather than domain-specific text, showing that the effect extends to behavioral post-training at 1B scale.

B.4.2 Upstream Dropout and Replay at 1B Parameters

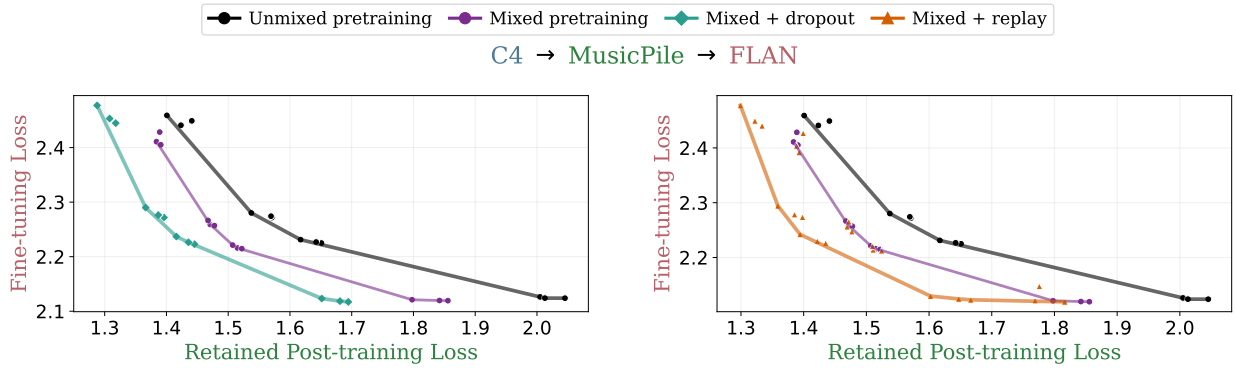


Figure 10: Replay and dropout provide complementary gains on top of mixed pretraining at 1B. Companion to Figure 4 at 1B scale. The left panel compares unmixed pretraining, mixed pretraining, and mixed pretraining + dropout; the right panel compares unmixed pretraining, mixed pretraining, and mixed pretraining + replay. As at 135M scale, both dropout and replay further shift the fine-tuning–retention frontier beyond mixed pretraining alone, indicating that these post-training interventions provide complementary gains at the larger scale as well.

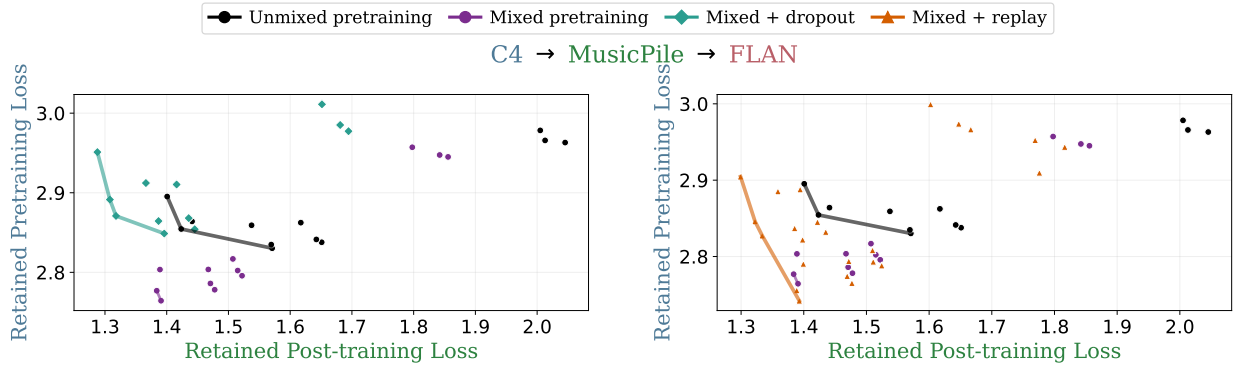


Figure 11: Dropout and replay on top of mixed pretraining, broader pretraining retention (1B). Companion to Figure 5 at 1B scale. The same Stage 2 sweeps are plotted against retained pretraining loss on C4. At this scale, dropout at the rate we swept degrades C4 retention relative to mixed pretraining alone, which we attribute to insufficient tuning of the dropout rate; replay continues to preserve and often improves C4 retention, consistent with its role of keeping general-domain data present during Stage 2.

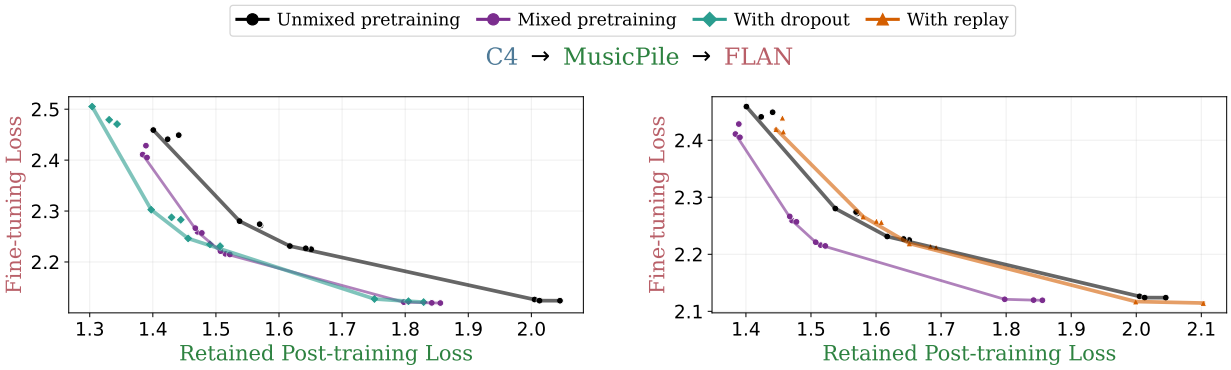


Figure 12: Dropout and replay applied without pretraining-time mixing (1B). Companion to Figure 6 at 1B scale. Each panel applies dropout or replay on top of unmixed pretraining ($\lambda = 0$), with the mixed-pretraining frontier shown for reference. At 1B, dropout continues to shift the fine-tuning–retention frontier relative to the unmixed baseline, while replay alone has a weaker effect, consistent with our observation in the main text that the relative strength of replay as a frontier-shifting intervention diminishes at this scale.

C Theoretical Analysis

C.1 Preliminaries and Setup

Model and data distribution We consider a two-layer linear network $\theta = \boldsymbol{W_1 W_2 x}$ on a series of regression problems using the squared loss. In particular, the problems take the form of $\mathcal{L}_t(\theta) = \boldsymbol{E}_{x\sim\mathcal{D}_t}[||\theta x - \boldsymbol{A}^t x||_2^2]$, where $t \in \{\text{pre}, \text{post}, \text{ft}\}$ indexes the. Here, $\mathcal{D}_t$ denotes the input distribution and the ground-truth outputs are generated as $\boldsymbol{A}^t \boldsymbol{X}$, where $\boldsymbol{X} \sim \mathcal{D}_t$. Following the analysis in Springer et al. (2025), we consider the singular values of $\boldsymbol{A}^t$ as the features for the training task t.

Assumption C.1 (Simultaneous Diagonalizability)**.** *There are orthonormal matrices $\boldsymbol{U}, \boldsymbol{V}$ such that for $t \in \{pre, post, ft\}$ we can write $\boldsymbol{A}^t = \boldsymbol{U}\boldsymbol{\Sigma_t}\boldsymbol{V}^\top$, where all the $\boldsymbol{\Sigma_t}$ are diagonal matrices.*

In order to model transfer and interference between the distributions, we will next specify a structure on the relationships between the different features. We first assume the presence of *invariant features*, capturing common linguistic capabilities that are broadly applicable across domains and tasks. Across these definitions, we assume a consistent ordering of the singular values (although the exact ordering of the singular values may shift).

Definition C.2 (Invariant Features)**.** *For $i \in [1, n-2k]$, we have that $(\boldsymbol{\Sigma}_{pre})_{ii} = (\boldsymbol{\Sigma}_{post})_{ii} = (\boldsymbol{\Sigma}_{ft})_{ii}$.*

In addition to these highly general features, we also consider the features through which the model may learn more domain specific information. We consider that such specialization can arise from either via a

Definition C.3 (Inconsistent Features)**.** *We define feature (indexed by i) to be inconsistent if we have that $(\boldsymbol{\Sigma}_{post})_{ii} > C(\boldsymbol{\Sigma}_{pre})_{ii}$*

Inconsistent features therefore incur a tradeoff between reducing loss on $\mathcal{D}_{\text{post}}$ and preserving performance on $\mathcal{D}_{\text{pre}}$. Finally, we introduce *specialized features*, which do not incur such a tradeoff.

Definition C.4 (Specialized features)**.** *We consider feature i is **specialized** if we have that $(\boldsymbol{V}^\top)_i \boldsymbol{x} = 0$ and that $(\boldsymbol{\Sigma}_{post})_{ii} > 0$. For simplicity we will assume that all specialized features take the same value of $(\boldsymbol{\Sigma}_{post})_{ii} = \beta$*

Intuitively, $\mathcal{D}_{\text{pre}}$ and $\mathcal{D}_{\text{ft}}$ have no covariance along the specialized feature directions. As we will show, this results in gradient steps taken along them causing no interference along these directions. However, as a result of their zero-covariance, these features are also impossible to learn without explicitly seeing the post-training data.

Downstream Tuning Task We consider that the downstream tuning task is relatively more similar to the pretraining task than the post-training task. As such, we consider that the inputs are sampled according to $x \sim \mathcal{N}(0, \boldsymbol{I}_{n-k})$ (i.e. it doesn't not activate the specialized features). As previously, we have that the singular values corresponding to the invariant features remain constant. However, the we have that the inconsistent features are misaligned between $\boldsymbol{A}^{\text{spec}}$ and $\boldsymbol{A}^{\text{ft}}$, in particular that $\sigma_i^{\text{ft}} = C_{\text{ft}}\sigma_i^{\text{spec}}$.

Mixed Training We parameterize the mixed distribution by a parameter α and train on the distribution $\mathcal{D}_{\alpha,\text{mixed}} = (1-\alpha)\mathcal{D}_{\text{pre}} + \alpha\mathcal{D}_{\text{sp}}$. As all distributions in our setting have mean zero, we have that the covariance matrix of this Gaussian mixture model is $(1-\alpha)\boldsymbol{\Sigma}_{\text{pre}} + \alpha\boldsymbol{\Sigma}_{\text{sp}}$.

Assumption C.5 (Invariant Features are High Magnitude)**.** *We consider that the invariant features are higher magnitude than the specialized features and the inconsistentfeatures, concretely:*

$$\sigma_i^{pre} > \sigma_j^{pre}$$

$\forall i \in [0, d_{invariant}]$ and $\forall j \in (d_{invariant}, n)$. We make a similar assumption on the relationship between the invariant features and the specialized features, concretely:

$$\sigma_i^{pre} > \sigma_j^{spec}$$

$\forall i \in [0, d_{invariant}]$ and $\forall j \in (d_{invariant}, n)$. This intuitively encodes that the invariant features have the highest salience in the data.

Assumption C.6 (Sufficient Specialized Mixing). *We assume that there exists* $\alpha > 0$

$$\alpha\beta > (1-\alpha)\sigma_i^{pre} + \alpha\sigma_i^{spec} \forall i \in [d_{invariant}, d_{invariant+k}]$$

Intuitively, Assumption C.6 suggests that there exists a mixing ratio such that the mixing specialized features become more salient than the inconsistent features. However, this mixing ratio need not be high if the strength of the specialized feature is high in the covariance of $\mathcal{D}_{\text{post}}$.

C.2 Analysis of Initial Pretraining

Here, we will study the dynamics of the pretraining stage. We first introduce an important result on the sequential learning dynamics of features in two layer linear networks (Gidel et al., 2019). For a given pretraining task where $\boldsymbol{X}, \boldsymbol{Y}$ represent the inputs and outputs, respectively. For a given task, we define that $\boldsymbol{\Sigma}_{xy} = \frac{1}{n}\boldsymbol{X}^\top \boldsymbol{Y}$ and $\boldsymbol{\Sigma}_x = \frac{1}{n}\boldsymbol{X}^\top \boldsymbol{X}$. We will write that $\boldsymbol{\Sigma}_{xy} = \sum_{i=1}^{R_{xy}} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$, where R_{xy} is the rank of $\boldsymbol{\Sigma}_{\boldsymbol{xy}}$. We will also assume that

Assumption C.7 (Joint Decomposition). *There exist orthogonal matrices* $\boldsymbol{U}, \boldsymbol{V}$ *such that*

$$\boldsymbol{\Sigma}_{xy} = \boldsymbol{U}\boldsymbol{D}_{xy}\boldsymbol{V}^\top, \boldsymbol{\Sigma}_{xx} = \boldsymbol{U}\boldsymbol{D}_{xx}\boldsymbol{U}^\top \tag{1}$$

and we will denote the singular values of $\boldsymbol{\Sigma}_{xy}$ *as* $\sigma_1, ..., \sigma_{R_{xy}}$ *and that the diagonal entries of* $\boldsymbol{D}_{xx}$ *as* $\lambda_1, ..., \lambda_{R_x} = 1$. *We*

Next, we will characterize the initialization scale of model before pretraining. Following (Springer et al., 2025), we have the following initialization:

Assumption C.8 (Pretrained Initialization Scale). *Let* $(\boldsymbol{W}_1(0), \boldsymbol{W}_2(0))$ *be the parameters at initialization. Then we have that* $\boldsymbol{W}_1(0) = \boldsymbol{W}_2(0) = \exp(-\mathcal{T})\boldsymbol{I_d}$.

Essentially, Assumption C.7 requires that the model parameters are close, but not exactly 0 which yields *sequential feature learning*. We next explicitly re-state the result from Gidel et al. (2019).

Theorem C.9 (Sequential Learning of Features Gidel et al. (2019); Springer et al. (2025)). *Suppose* $\boldsymbol{W}_1, \boldsymbol{W}_2$ *obey the initialization in Assumption C.8 and the pretraining task obeys Assumption C.7. Then there exist times* $t_1, ..., t_r$ *such that*

$$||\boldsymbol{W}_1(t_i) - \boldsymbol{U}(\Sigma_{:i})^{\frac{1}{2}}||_F \leq \exp(-C\tau)$$
$$||\boldsymbol{W}_2(t_i) - (\Sigma_{:i})^{\frac{1}{2}}\boldsymbol{V}^\top||_F \leq \exp(-C\tau)$$

Where $\Sigma_{:i}$ *is defined to be* $diag(\sigma_1, ...\sigma_i, 0, ...0)$, *equivalently the rank* i *approximation of* $diag(\sigma_1, ..., \sigma_{R_{xy}})$.

Conceptually, Theorem C.9 demonstrates that during the pretraining process, $\boldsymbol{W}_1\boldsymbol{W}_2$ learn features in decreasing order of their of the singular value of Σ_{xy}. Next, we will apply this result in order to compare the features learned during mixed and non-mixed pretraining.

Theorem C.10 (Only Mixing Learns Specialized Features). *Let* $\theta^{gen}(t) = \boldsymbol{W}_1^{gen}(t)\boldsymbol{W}_2^{gen}(t)$ *be the parameters learned when pretraining only on* $\mathcal{D}_{gen}$ *and* $\theta^{mixed}(t) = \boldsymbol{W}_1^{mixed}(t)\boldsymbol{W}_2^{mixed}(t)$. *Denote* $\boldsymbol{u}_{spec}, \boldsymbol{v}_{spec}$ *to be the right and left singular values corresponding to the specialized feature. Then, there exists a time* t *such that*

$$||\boldsymbol{W}_1^{(unmixed)} - \boldsymbol{U}(\boldsymbol{\Sigma}^{(unmixed)})^{\frac{1}{2}}||_F \leq \exp(-C\tau)$$
$$||\boldsymbol{W}_2^{(unmixed)} - (\boldsymbol{\Sigma}^{(unmixed)})^{\frac{1}{2}}\boldsymbol{V}^\top||_F \leq \exp(-C\tau)$$
$$||\boldsymbol{W}_1^{(unmixed)} - \boldsymbol{U}(\boldsymbol{\Sigma}^{(unmixed)})^{\frac{1}{2}}||_F \leq \exp(-C\tau)$$
$$||\boldsymbol{W}_2^{(unmixed)} - (\boldsymbol{\Sigma}^{(unmixed)})^{\frac{1}{2}}\boldsymbol{V}^\top||_F \leq \exp(-C\tau)$$

where $\boldsymbol{\Sigma}_{mixed} = diag(\sigma_1^{inv}, ...\sigma_k^{inv}, \boldsymbol{0}_k, \alpha\beta, ..., \alpha\beta)$ *and* $\boldsymbol{\Sigma}_{unmixed} = diag(\sigma_1^{inv}, ..., \sigma_1^{inc}, ..., \sigma_k^{inc}, \boldsymbol{0}_k)$

Proof. This is a relatively straightforward application of Theorem C.9. Denote $(\boldsymbol{X}^{(\text{mixed})}, \boldsymbol{Y}^{(\text{mixed})})$ as the data used for mixed pretraining and $\boldsymbol{\Sigma}_{xy}^{(\text{mixed})} = \frac{1}{n}(\boldsymbol{X}^{(\text{mixed})})^\top \boldsymbol{Y}^{(\text{mixed})}$. We have that $\boldsymbol{\Sigma}_{xy}^{(\text{mixed})} = (1-\alpha)\boldsymbol{\Sigma}_{xy}^{\text{pre}} + \alpha\boldsymbol{\Sigma}_{xy}^{\text{spec}}$. By Theorem C.9, we have that the features are learned in order of the singular values of $\boldsymbol{\Sigma}_{\boldsymbol{xy}}$. By Assumptions C.5 and C.6, we have that the top $n-k$ singular values of $\boldsymbol{\Sigma}_{xy}^{(\text{mixed})}$ are the $n-2k$ shared features and the k specialized features. Define $\boldsymbol{\Sigma}_{:n-k}^{(\text{mixed})} = \text{diag}(\sigma_1^{\text{inv}}, ...\sigma_k^{\text{inv}}, \mathbf{0}_k, \alpha\beta, ..., \alpha\beta)$. Applying Theorem C.9, we have that

$$||\boldsymbol{W}_1^{(\text{mixed})} - \boldsymbol{U}(\boldsymbol{\Sigma}_{:n-k}^{(\text{mixed})})^{\frac{1}{2}}||_F \leq \exp(-C\tau)$$
$$||\boldsymbol{W}_2^{(\text{mixed})} - (\boldsymbol{\Sigma}_{:n-k}^{(\text{mixed})})^{\frac{1}{2}}\boldsymbol{V}^\top||_F \leq \exp(-C\tau)$$

Repeating this analysis for unmixed training, we have that the top $n-k$ singular values of $\boldsymbol{\Sigma}_{xy}^{(\text{gen})}$ are the $n-2k$ shared features are the k inconsistent features. We can define $\boldsymbol{\Sigma}_{:n-k}^{(\text{unmixed})} = \text{diag}(\sigma_1^{\text{inv}}, ...\sigma_k^{\text{inv}}, \sigma_1^{\text{shared-mis}}, ..., \sigma_1^{\text{shared-mis}}, \mathbf{0}_k)$. Similarly, by applying Theorem C.9, we have that

$$||\boldsymbol{W}_1^{(\text{unmixed})} - \boldsymbol{U}(\boldsymbol{\Sigma}_{:n-k}^{(\text{unmixed})})^{\frac{1}{2}}||_F \leq \exp(-C\tau)$$
$$||\boldsymbol{W}_2^{(\text{unmixed})} - (\boldsymbol{\Sigma}_{:n-k}^{(\text{unmixed})})^{\frac{1}{2}}\boldsymbol{V}^\top||_F \leq \exp(-C\tau)$$

□

Intuitively, our result in Theorem C.9 demonstrates that mixing $\mathcal{D}_{\text{post}}$ during pretraining results in a pretrained initialization that has different features. Mixing learns the specialized features, while not mixing learns only the inconsistent features. In the following, we will examine the impact that these different features have on the retention of $\mathcal{D}_{\text{post}}$ during subsequent training.

C.3 Analysis of Post-Training

We now study the dynamics of the post-training process. To formalize the post-training process, we first examine the dynamics beginning from the idealized pretraining initialization (as performed by Springer et al. (2025)). We perform the post-training stage on the regularized loss $\boldsymbol{E}[||\theta x - \boldsymbol{A}^{\text{sp}}x||_F^2] + \lambda||\theta - \theta_0||_F^2$. Observe that because $x \sim \mathcal{N}(0, \boldsymbol{I}_d)$, this is equivalent to $||\theta - \boldsymbol{A}^{\text{sp}}||_F$. We follow the assumptions on the regularity of fine-tuning established in Springer et al. (2025). Our analysis imposes certain

Assumption C.11 (Bound on Parameters Throughout Training)**.**

$$||\hat{\boldsymbol{W}}_1^{(\textit{mixed})}||_{op} \leq \sqrt{\Gamma}$$
$$||\hat{\boldsymbol{W}}_1^{(\textit{mixed})}||_{op} \leq \sqrt{\Gamma}$$
$$||\hat{\boldsymbol{W}}_1^{(\textit{unmixed})}||_{op} \leq \sqrt{\Gamma}$$
$$||\hat{\boldsymbol{W}}_1^{(\textit{unmixed})}||_{op} \leq \sqrt{\Gamma}$$

Moreover, we assume that the regularization strength and the learning rates are likewise bounded.

Assumption C.12 (Bound on Learning Rate)**.**

$$4\eta(\lambda+2)\Gamma < 1$$

Idealized Pretraining Initialization We denote the ideal initialization parameters for the mixed and unmixed cases $(\hat{\boldsymbol{W}}_1(0), \hat{\boldsymbol{W}}_2(0))$.

$$\hat{\boldsymbol{W}_1^{(\text{mixed})}}(0) = \boldsymbol{U}(\Sigma_{:n-k}^{\text{mixed}})^{\frac{1}{2}}$$
$$\hat{\boldsymbol{W}_2^{(\text{mixed})}}(0) = (\Sigma_{:n-k}^{\text{mixed}})^{\frac{1}{2}}\boldsymbol{V}^\top$$

Similarly, we have the following idealized initialization for the unmixed initialization

$$\hat{\boldsymbol{W}_1^{(\text{unmixed})}}(0) = \boldsymbol{U}(\Sigma^{\text{unmixed}}_{:n-k})^{\frac{1}{2}}$$
$$\hat{\boldsymbol{W}_2^{(\text{unmixed})}}(0) = (\Sigma^{\text{unmixed}}_{:n-k})^{\frac{1}{2}}\boldsymbol{V}^\top$$

In the idealized setting, we can track the evolution of each singular value independently. In particular, we have the following update rules as derived in Springer et al. (2025) (where we denote σ_i^{spec} as the i-th singular value of $\boldsymbol{A}^{\text{spec}}$ and likewise for $\sigma_i^{(\text{un})\text{mixed}}(t)$ as the i-th singular value at step t.In what follows, we will suppress the superscript for compactness:

$$\sigma_i(t+1) = \sigma_i(t) + 2\eta\sigma_i(t)(\eta\sigma_i(t)^2 - (\sigma_{\text{spec},i})^2) + 2\eta\lambda(\sigma_i(t)^2 - \sigma_i(0)^2) \tag{2}$$

As a result, note that when $\sigma_i^{(\text{un})\text{mixed}}(0) = 0$, $\sigma_t^{(\text{un})\text{mixed}}(t) = 0$ for all t.

Next,we will study the dynamics of the non-zero singular values (Lemma A.11 Springer et al. (2025). We will assume that post-training is performed for a sufficient number of steps.

Assumption C.13 (Sufficient Post-Training Steps)**.** *We have that the number of post-training steps (denoted by K) satisfies $K \geq \frac{1}{\lambda c_{min}} \log \frac{100\Gamma}{\epsilon}$, for a constant ϵ and where $c_{min} = \min\{(\boldsymbol{\Sigma}_{post})_{ii} | \boldsymbol{\Sigma}_{post})_{ii} \neq 0\}$ – that is the minimum, non-zero singular value.*

Given these technical conditions, we now state a general result (adapted for our setting from Springer et al. (2025)).

Lemma C.14. *When training on $\mathcal{D}_{post}$ with infinite batch size from the ideal pretraining initialization and taking sufficient number of steps K, for all $i \in rank(\theta_n(0))$, we have that*

$$|(\boldsymbol{U}^\top \hat{\theta}^{(un)mixed}(t)\boldsymbol{V})_{ii} - (\boldsymbol{\Sigma}_{post})_{ii}| \leq \epsilon \tag{3}$$

where $\boldsymbol{\Sigma}_{post}$ is such that $\boldsymbol{A}_{post} = \boldsymbol{U}\boldsymbol{\Sigma}_{post}\boldsymbol{V}^\top$.

Across the settings, we assume that K is sufficiently large such that $\epsilon < \frac{(n-k)(\alpha-1)\min_i \sigma^{\text{spec}_i}}{4K\beta+2(\alpha-1)}$ Now, we will define the matrices $\boldsymbol{\Sigma}^{\text{shared,post}} = \text{diag}(\sigma_1^{\text{inv}}, ...\sigma_{n-2k}^{\text{inv}}, \sigma^{\text{spec}}_{d_{\text{invariant}}+1}, ..\sigma^{\text{spec}}_{d_{\text{invariant}}+k}, \boldsymbol{0}_k)$ and $\boldsymbol{\Sigma}^{\text{spec,post}} = \text{diag}(\sigma_1^{\text{inv}}, ...\sigma_{n-2k}^{\text{inv}}, \boldsymbol{0}_k, \sigma^{\text{spec}}_{d_{\text{invariant}}+k+1}, ..\sigma^{\text{spec}}_{d_{\text{invariant}}+2k})$, Intuitively, $\boldsymbol{\Sigma}^{\text{shared,post}}$ lowers loss on $\mathcal{D}_{\text{post}}$ by shifting the values on the shared features, while $\boldsymbol{\Sigma}^{\text{spec,post}}$ accomplishes this by modifying the unique features. We are now ready to state the main theorem.

Theorem C.15 (Post-training on θ^{mixed} versus θ^{unmixed})**.** *Let $\theta^{mixed}(K)$ denote the parameters after training the idealized unmixed initialization starting for K steps and let $\theta^{mixed}(K)$ be the same starting from the idealized mixed checkpoint. Then we have*

$$||\boldsymbol{U}^\top\theta^{mixed}\boldsymbol{V} - \boldsymbol{\Sigma}^{spec,post}||_{op} \leq \epsilon$$
$$||\boldsymbol{U}^\top\theta^{unmixed}\boldsymbol{V} - \boldsymbol{\Sigma}^{shared,post}||_{op} \leq \epsilon$$

Proof. This theorem follows by noting that under the idealized pretraining initialization any singular value that is 0 at initialization remains that way during the entire optimization trajectory. Note that the 0 singular values of $\boldsymbol{U}^\top\theta^{\text{mixed}}\boldsymbol{V}$ coincide with $\boldsymbol{\Sigma}^{\text{spec, post}}$ (the specialized features) and likewise $\boldsymbol{U}^\top\theta^{\text{unmixed}}\boldsymbol{V}$ coincide with $\boldsymbol{\Sigma}^{\text{shared, post}}$ (the inconsistent features).

This implies that we have

$$\max_{i\in[1,n]} |(\boldsymbol{U}^\top\theta^{\text{mixed}}\boldsymbol{V})_{ii} - (\boldsymbol{\Sigma}^{\text{spec,post}})_{ii}| \leq \max_{i\in\text{rank}(\theta)} |(\boldsymbol{U}^\top\theta^{\text{mixed}}\boldsymbol{V})_{ii} - (\boldsymbol{\Sigma}^{\text{spec,post}})_{ii}|$$
$$\max_{i\in[1,n]} |(\boldsymbol{U}^\top\theta^{\text{unmixed}}\boldsymbol{V})_{ii} - (\boldsymbol{\Sigma}^{\text{shared,post}})_{ii}| \leq \max_{i\in\text{rank}(\theta)} |(\boldsymbol{U}^\top\theta^{\text{mixed}}\boldsymbol{V})_{ii} - (\boldsymbol{\Sigma}^{\text{shared,post}})_{ii}|$$

Now, applying the result from Lemma C.14 yields the desired claim. □

C.4 Analysis of Downstream Adaptation

In the previous section, we characterized the impact of post-training from a mixed versus an unmixed initialization, demonstrating that different features are used to minimize the loss on $\mathcal{D}_{\text{post}}$. In this section, we study how these different features impact the ultimate . We first establish that the singular values corresponding to directions in which there is no covariance remain unchanged throughout the downstream fine-tuning stage.

Lemma C.16. *Consider performing downstream unregularized fine-tuning on $\mathcal{D}_{ft}$. If $x \sim \mathcal{D}_{ft}$ has 0 covariance along a singular direction, the corresponding singular value remains unchanged throughout downstream adaptation.*

Proof. To see this, note that the gradient updates for $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ take the following form:

$$\boldsymbol{W}_1(k+1) = \boldsymbol{W}_1(k) - 2\eta(\boldsymbol{W}_1(k)\boldsymbol{W}_2(k) - \boldsymbol{A}^{\text{spec}})\Sigma_x \boldsymbol{W}_2^\top$$
$$\boldsymbol{W}_2(k+1) = \boldsymbol{W}_2(k) - 2\eta \boldsymbol{W}_1(k)\Sigma_x(\boldsymbol{W}_1(k)\boldsymbol{W}_2(k) - \boldsymbol{A}^{\text{spec}})$$

Here, we have that $\Sigma_x$denotes the covariance of the input data x. Thus, along any singular direction in which the data has 0 variance, the $\boldsymbol{\Sigma}_x$ term will project the gradient to 0. Thus, the singular values on such directions must also remain unchanged. □

We consider performing downstream adaptation by taking steps using unregularized gradient descent on $\mathcal{D}_{\text{ft}}$ and show the following result.

Theorem C.17. *Consider performing K steps of gradient descent on the downstream finetuning dataset beginning from the intializations $\theta^{\text{post, mixed}}(K)$ and let $\theta^{\text{FT,mixed}}(K)$, $\theta^{\text{FT,unmixed}}(K)$ $\theta^{\text{post, unmixed}}(K)$ denote the final parameters. Let $\Delta_{\text{unmixed}} = \mathcal{L}(\theta^{\text{FT, mixed}}; \mathcal{D}_{\text{spec}}) - \mathcal{L}(\theta^{\text{post, mixed}}(K); \mathcal{D}_{\text{spec}})$ and likewise $\Delta_{\text{mixed}} = \mathcal{L}(\theta^{\text{FT, unmixed}}; \mathcal{D}_{\text{spec}}) - \mathcal{L}(\theta^{\text{post,unmixed}}(K); \mathcal{D}_{\text{spec}})$. Then we have that $\Delta_{\text{unmixed}} \geq \Delta_{\text{mixed}}$.*

Proof. As the invariant features take the same values, they will not move during the downstream adaptation. Moreover, due to the Lemma C.16, we also have that the specialized features will not change during the the downstream fine-tuning. This implies that $\Delta_{\text{unmixed}} = 0$. Next we will examine the changes induced by downstream training on the unmixed models. Observe that due to the full-rankedness of covariance of $\mathcal{D}_{\text{spec}}$, we have that the $\mathcal{L}(\theta; \mathcal{D}_{\text{spec}}) = ||\theta - \boldsymbol{A}^{\text{spec}}||_F^2$. By applying Lemma C.14 We define the following matrices $\Sigma_{\text{FT}}^{(\text{unmixed})} = \text{diag}(\sigma_1^{\text{invariant}}, ..., \sigma_{d_{\text{invariant}}}^{\text{invariant}}, \sigma_{d_{\text{invariant}}+1}^{\text{ft}}, ...\sigma_{\text{d}_{\text{invariant}}+k}^{\text{ft}} \mathbf{0}_k)$ and note that Lemma C.14 gives us that

$$||\boldsymbol{U}^\top \theta^{(\text{unmixed})_{\text{FT}}} \boldsymbol{V} - \Sigma_{\text{FT}}^{(\text{unmixed})}||_{\text{op}} \leq \epsilon$$

Likewise, we have that

$$||\boldsymbol{U}^\top \theta^{(\text{unmixed})_{\text{post}}} \boldsymbol{V} - \Sigma_{\text{FT}}^{(\text{unmixed})}||_{\text{op}} \leq \epsilon$$

The loss function we use here is simply the squared difference of the singular values. Thus,we can upper bound:

$$\mathcal{L}(\theta_{\text{post}}^{\text{unmixed}}; \mathcal{D}_{\text{spec}}) \leq k(\beta + \epsilon)^2 + (n-k)\epsilon^2$$

and likewise lower bound

$$\mathcal{L}(\theta_{\text{ft}}^{\text{unmixed}}; \mathcal{D}_{\text{spec}}) \geq k(\beta - \epsilon)^2 + (n-k)((\alpha - 1) - \epsilon)^2$$

Then,we can lower bound $\Delta_{\text{unmixed}} \geq k[(\beta+\epsilon)^2 - (\beta-\epsilon)^2] + (n-k)[(\alpha - 1 - \epsilon)^2 - \epsilon^2]$. From the upper bound of ϵ, we have that this implies $\Delta_{\text{mixed}} \geq 0$. Hence, we have $\Delta_{\text{unmixed}} \geq \Delta_{\text{mixed}}$. □