REPLAY ON DEMAND: AN EMERGENT CURRICULUM FOR BALANCING ADAPTATION AND FORGETTING IN CONTINUED PRETRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Continued pretraining enables language models to adapt to new domains and knowledge, but often at the cost of forgetting previously acquired capabilities. Replay can mitigate this trade-off, but fixed replay mixtures allocate training independently of the model's actual retention needs. This is particularly limiting because forgetting varies across capabilities and data sources and evolves throughout training. We introduce *Replay on Demand* (RoD), which instead derives the replay allocation from the model's learning dynamics. RoD jointly prioritizes adaptation samples by their remaining learning potential and replay samples by their observed forgetting. Their competition for a shared training budget yields an online curriculum that determines what to train on at each step. Across models, scales, and adaptation domains, RoD reaches or improves upon the adaptation–forgetting frontier of tuned fixed-replay baselines and model merging without prescribing a replay allocation in advance. Replay concentrates on sources that are more vulnerable to forgetting and dynamically increases and redistributes as forgetting emerges during training. Together, our results show that replay can be allocated online from the model's evolving state, targeting *what is needed, when it is needed.*

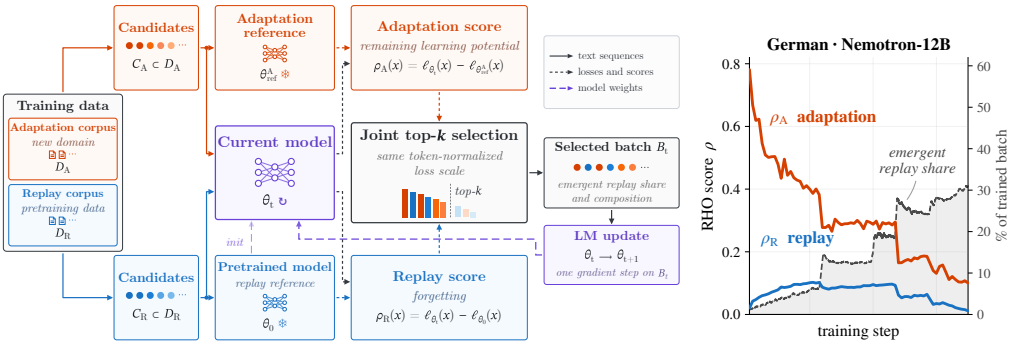


Figure 1: **RoD dynamically balances adaptation and replay through joint data selection.** Adaptation candidates are scored by their remaining learning potential and replay candidates by their forgetting. Joint top-k selection lets both compete for a shared training budget, dynamically determining the replay share and composition. As adaptation learning potential decreases and forgetting emerges, replay becomes increasingly competitive and receives a larger share of the training budget.

1 INTRODUCTION

Continued pretraining (CPT) has become a practical route for adapting general-purpose language models to specialized domains, languages, and applications (Gururangan et al., 2020; Parmar et al., 2024). However, this specialization comes at a cost: learning a new distribution can degrade capabilities acquired during pretraining. CPT therefore faces an inherent trade-off between *adaptation* and *retention*: a successful method should adapt to the new distribution while minimizing forgetting of the original pretraining distribution.

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

A common strategy for controlling this trade-off is *replay*, which mixes samples from the original pretraining distribution into the adaptation data (Abbes et al., 2026; Roth et al., 2024). Yet replay turns this trade-off into a resource-allocation problem. Under a fixed training budget, replay displaces adaptation data, while maintaining both requires additional compute. Replay should therefore be allocated where it provides the greatest retention benefit. This is challenging because forgetting is heterogeneous. Different capabilities and parts of the pretraining distribution degrade to different degrees during adaptation (Thede et al., 2026; Çağatay Yıldız et al., 2025). Fixed replay mixtures ignore this heterogeneity, potentially spending compute on well-retained knowledge while leaving more vulnerable knowledge insufficiently protected. Moreover, retention needs evolve throughout training. An effective replay curriculum should therefore allocate replay where forgetting occurs and adapt this allocation as retention needs change.

We propose *Replay on Demand* (RoD), which allocates training compute based on the model's current learning and forgetting states. RoD builds on reducible-loss data selection (Mindermann et al., 2022) and lets adaptation and replay examples compete for a shared training budget. Adaptation examples are prioritized by how much remains to be learned, while replay examples are prioritized by how much has been forgotten relative to the pretrained model. This competition gives rise to an online data curriculum. When pretrained knowledge is well retained, adaptation dominates training. As forgetting emerges, affected replay examples become more competitive and enter the training batch. RoD thereby concentrates replay on the parts of the pretraining distribution that degrade, dynamically adjusting its amount and composition throughout training. The resulting curriculum adapts to the model and adaptation corpus without requiring a predefined replay share or allocation.

We evaluate RoD across legal and German continued pretraining using the Nemotron and Qwen model families at scales ranging from 4B to 35B parameters. Across these settings, RoD reaches or improves upon the adaptation–forgetting frontier of tuned fixed-replay CPT and model merging, without prescribing a replay allocation in advance. Our analysis shows that these gains reflect the intended demand-driven allocation: RoD directs replay toward sources that exhibit greater forgetting, avoids unnecessary replay of well-retained ones, and dynamically increases and redistributes replay as retention needs emerge during training. Together, these results show that adapting replay to the model's evolving retention needs provides a principled way to balance adaptation and retention by replaying what is needed, when it is needed.

2 Related Work

Continual pretraining and replay. CPT adapts pretrained language models to new data while aiming to preserve previously acquired capabilities (Gururangan et al., 2020; Çağatay Yıldız et al., 2025). Replay of pretraining data is a common strategy for mitigating forgetting (Ibrahim et al., 2024; Parmar et al., 2024; Roth et al., 2024), making the allocation of training compute between adaptation and retention a central design choice. Recent work has studied this trade-off through scaling laws for source–target mixture ratios (Gu et al., 2024; Que et al., 2024), adaptive data mixtures (Chen et al., 2025; Luo et al., 2025; Yang et al., 2026), and replay mechanisms that select or schedule past data based on their utility or the model's learning state (Atreya et al., 2026; Feng et al., 2026; Smith et al., 2024). Correspondingly, CPT evaluation considers both adaptation and retention rather than target-domain performance alone (Jin et al., 2022; Li et al., 2025). We build on this perspective but ask whether this allocation can emerge from the model's learning and forgetting.

Reducible-loss data selection. Reducible Holdout Loss (RHO) was introduced for data-efficient training, prioritizing examples according to how much of their current loss remains reducible relative to a reference model (Mindermann et al., 2022). Subsequent work extends reference-model-guided selection to sequences (Thirukovalluru et al., 2024) and tokens (Lin et al., 2024) in language-model pretraining. Related work uses model-dependent signals to adapt pretraining mixtures (Wang et al., 2026), while CSReL applies reducible-loss selection to replay examples in continual learning (Tong et al., 2025). These methods determine which data to train on based on the model's state, but do not jointly allocate compute between adaptation and retention.

Positioning RoD. RoD connects adaptive CPT and model-dependent data selection by placing adaptation and replay examples in a shared competition. Unlike methods optimizing a source–target mixture (Chen et al., 2025; Gu et al., 2024; Luo et al., 2025; Que et al., 2024; Yang et al., 2026)

or select replay within a dedicated mechanism (Atreya et al., 2026; Feng et al., 2026; Smith et al., 2024; Tong et al., 2025), RoD jointly determines replay amount and composition from the relative utility of adaptation and retention candidates. Its reference models encode these objectives: the adaptation reference estimates learnability, while the pretrained model measures degradation. The adaptation–retention trade-off thus emerges from the model's evolving state rather than a predefined mixture or separate replay mechanism. Controlled fixed-allocation baselines isolate this distinction by prescribing the replay allocation while retaining model-dependent selection (section 4.3).

3 METHOD

We introduce *Replay on Demand* (RoD), an online data-selection method that dynamically allocates training compute between adaptation and replay. As illustrated in fig. 1, RoD scores adaptation and replay examples with source-specific reducible losses and lets them compete for a shared training budget. The resulting selection determines both *how much* and *what* to replay throughout training.

3.1 PROBLEM SETUP

Let θ_0 denote a pretrained language model that we adapt to a new data distribution $\mathcal{D}_A$ while retaining performance on its pretraining distribution $\mathcal{D}_R$. During continual pretraining, θ_t denotes the model after t updates. We construct candidate sets

$$C_A \subset \mathcal{D}_A, \qquad C_R \subset \mathcal{D}_R, \tag{1}$$

and select a training batch B_t of k examples from their union. We denote the token-normalized language-modeling loss of model θ on an example x by $\ell_\theta(x)$.

Rather than fixing the fraction of adaptation and replay data in B_t beforehand, RoD lets candidates from both sources compete for the same training budget based on their current utility to the model.

3.2 SOURCE-SPECIFIC REDUCIBLE LOSS

RoD builds on Reducible Holdout Loss (RHO) (Mindermann et al., 2022), which scores examples by the difference between their current loss and the loss of a reference model:

$$\rho(x;\theta_t) = \ell_{\theta_t}(x) - \ell_{\theta_{\text{ref}}}(x). \tag{2}$$

The reference loss estimates how well an example can be explained, such that the difference captures the loss that remains reducible by further learning. RHO therefore prioritizes examples that the current model does not yet explain well, but that a suitable reference model does.

In continual pretraining, adaptation should target what remains to be learned, while replay targets what has been forgotten. We capture both with the same reducible-loss formulation, choosing the reference model according to the candidate source:

$$\rho(x;\theta_t) = \begin{cases} \rho_A(x;\theta_t) = \ell_{\theta_t}(x) - \ell_{\theta^A_{\text{ref}}}(x), & x \in C_A, \\ \rho_R(x;\theta_t) = \ell_{\theta_t}(x) - \ell_{\theta_0}(x), & x \in C_R. \end{cases} \tag{3}$$

where θ^A_{ref} is the adaptation reference model specialized to the adaptation distribution and the pretrained model θ_0 serves as the replay reference. Both scores therefore measure excess loss relative to a source-appropriate reference, providing a common scale for joint selection.

This construction follows from a source-specific approximation of the original RHO objective, which measures how much training on an example improves the posterior predictive likelihood of a holdout distribution. In CPT, the holdout contains both adaptation and retained pretraining data. Approximating its posterior predictive separately for each source yields θ^A_{ref} for adaptation and θ_0 for replay. A single reference trained on both would instead encode an adaptation–retention trade-off through its training mixture. We provide the full derivation and assumptions in section A.1.

Adaptation: remaining learning potential. For adaptation examples, θ^A_{ref} estimates the loss attainable after learning the adaptation distribution, without requiring the reference to retain the general capabilities of θ_0. The resulting ρ_A is high for examples that the current model does not yet explain well but the adaptation specialist does. As the model learns these examples, their scores decrease. Thus, ρ_A prioritizes remaining learning potential on $\mathcal{D}_A$.

Replay: forgetting from the pretrained state. For replay examples, the pretrained model θ_0 provides a natural reference for detecting degradation. Here, ρ_R directly measures the change in loss since the start of adaptation. Retained examples have $\rho_R(x;\theta_t) \approx 0$, whereas examples whose performance has degraded obtain positive scores. Replay therefore prioritizes knowledge in proportion to its currently observed forgetting.

3.3 Joint Adaptation and Replay Selection

Given candidate sets C_A and C_R, RoD scores each candidate according to its source and selects the globally highest-scoring k examples:

$$B_t = \mathrm{TopK}_{x \in C_A \cup C_R} \rho(x;\theta_t). \tag{4}$$

The model is then updated on B_t using the standard language-modeling objective. Under a local marginal-utility approximation, this selection allocates each training slot to candidates with the largest expected reduction in their source objective. The replay share thus emerges from the relative demand of adaptation and replay rather than a predefined ratio. We formalize this interpretation and its assumptions in section A.2.

We control the size of the candidate pool via a candidate multiplier m, with $|C_A| + |C_R| = mk$. Larger m allows more candidates to compete for each training slot, providing greater flexibility in selecting the batch at the cost of additional scoring. We use $m = 2$ by default and study its effect in Section 4.3.

Figure 1b illustrates how this competition produces a dynamic replay allocation. At the start of adaptation, $\theta_t = \theta_0$ and all replay scores are zero, such that adaptation examples dominate selection. As adaptation progresses, scores decrease for learned adaptation examples, while replay scores increase wherever pretrained knowledge degrades. Replay consequently becomes more competitive and receives a larger share of the training budget. Within the replay pool, examples with greater degradation similarly outrank well-retained examples. RoD thereby adapts both the replay share and replay composition to the model's evolving learning and forgetting state.

Practical implementation. Reference-model losses are fixed throughout continual pretraining and can therefore be precomputed and cached for both candidate sources. Each training step then requires one inference pass of the current model over the candidate pool for scoring, followed by a standard training pass over the selected batch. While RoD does not require equal candidate-pool sizes, we balance adaptation and replay candidates in our experiments so that $m = 2$ allows the selected batch to consist entirely of either source. We obtain θ^A_{ref} by training the pretrained model on $\mathcal{D}_A$ until convergence, without replay or other forgetting mitigation. Further algorithmic and implementation details are provided in sections A.4 and A.5. Experimental details are provided in section B, and computational requirements and limitations are discussed in section D. Our implementation builds on NeMo-RL (nem, 2025), with code for RoD and our experimental setup available in our anonymized repository at `https://anonymous.4open.science/r/replay-on-demand-7042/README.md`.

4 Experiments

We evaluate RoD across two model families and two adaptation domains, progressively testing the generality of our approach. We evaluate Nemotron-Nano-12B-v2 (NVIDIA, 2025) on Legal and German with access to its original pretraining data for replay, and Qwen3.5-9B (Qwen Team, 2026) on both domains to test generalization across model families and to proxy replay. For our cross-scale experiments in section 4.4, we additionally use Qwen3.5-4B as a smaller source model and transfer curricula constructed at smaller scale to Nemotron-30B (NVIDIA, 2026) and Qwen3.5-35B as larger target models. Full experimental details are provided in section B.

We compare RoD against the no-replay baseline, CPT with several fixed replay shares, and post-hoc model merging between the base and no-replay CPT models. RoD and fixed-replay CPT use the same learning-rate schedule and matched trained-token budgets, while RoD incurs additional candidate-scoring overhead (see section D.1). We additionally train a max-replay reference on the full available adaptation–replay mixture using approximately $2\times$ the trained-token budget, and therefore treat it as a higher-budget reference rather than part of the matched-budget comparison.

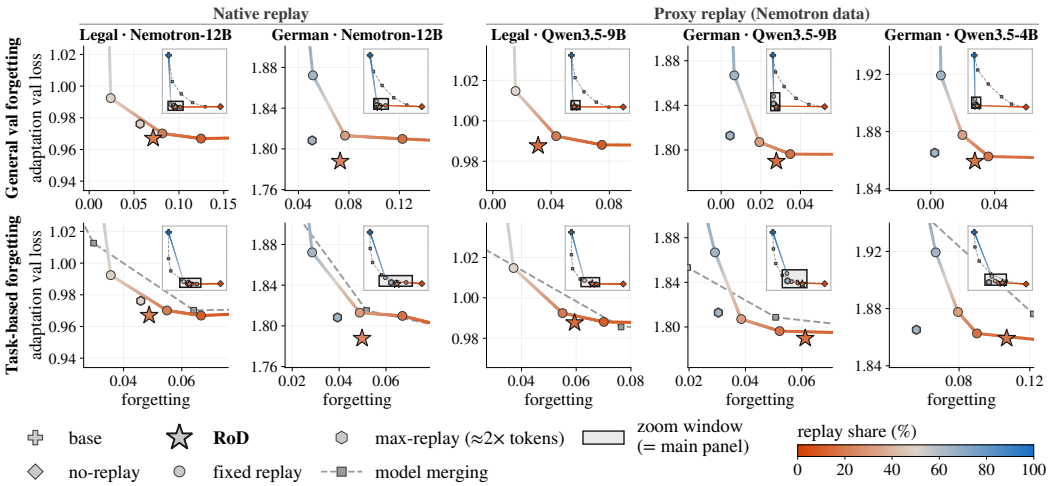


Figure 2: **RoD reaches or improves upon the adaptation–forgetting frontier.** Adaptation loss is shown against general validation-loss forgetting (top) and task-based forgetting (bottom); lower is better on both axes. Fixed-replay CPT traces the frontier as replay share varies, while RoD reaches or improves upon it without specifying a replay ratio in advance. Nemotron uses native replay from its pretraining data, whereas Qwen uses the Nemotron data as proxy replay because its pretraining data are unavailable. No-replay CPT and model merging provide additional baselines; max replay is a higher-budget reference trained with approximately $2\times$ the trained-token budget. Insets magnify the operating region around RoD and the strongest baselines.

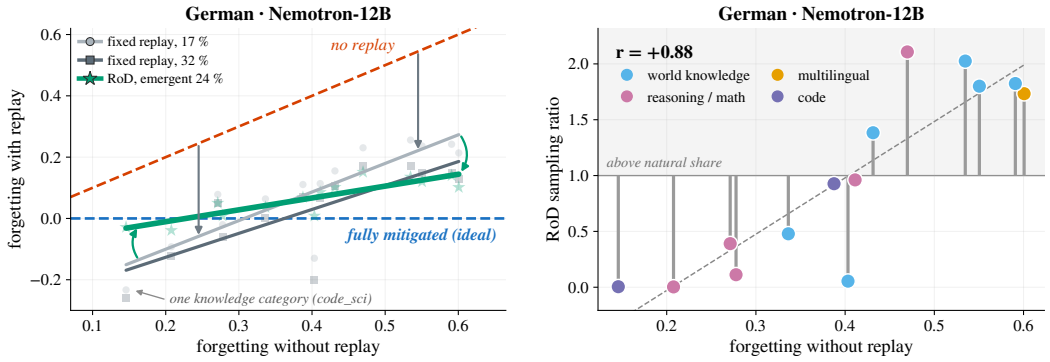
We measure adaptation by validation loss on held-out adaptation data and forgetting by the increase in held-out general validation loss relative to the base model. We complement this distribution-level evaluation with a capability-level perspective, using multiple-choice evaluations of four latent capability groups following the CapTrack taxonomy (Thede et al., 2026): parametric knowledge, reasoning and problem-solving, commonsense and robustness, and multilingual capabilities. Further details on data construction, training, baselines, and evaluation are provided in section B.

4.1 RoD improves the adaptation–forgetting frontier

We characterize the stability–plasticity trade-off through the adaptation–forgetting frontier in fig. 2, where desirable methods combine strong adaptation with low validation-loss and task-based forgetting. Fixed-replay CPT traces this frontier as replay varies: more replay reduces forgetting but limits target-domain adaptation. Since the preferred balance is setting-dependent, selecting a fixed replay ratio requires prior knowledge or a sweep over ratios. We use the full sweep as a reference frontier, noting that selecting its best point for comparison constitutes an oracle-like choice. Complete numerical results and fixed-replay baselines trained to convergence are provided in section C.1.

Does RoD improve the frontier with native replay? Across both Nemotron settings, RoD reaches or improves upon the fixed-replay frontier without specifying a replay ratio in advance. At matched adaptation on Legal, RoD reduces validation-loss forgetting by 43% relative to 10% fixed replay (0.125 to 0.071) and task-based forgetting from 6.7 to 4.9 points. On German, RoD maintains substantially stronger adaptation while closely matching the retention of the more replay-heavy 32% baseline. RoD also compares favorably with alternative strategies: model merging achieves stronger adaptation only at substantially higher forgetting, whereas max-replay training yields slightly better retention with approximately twice the trained-token budget. Despite this additional budget, RoD achieves better adaptation than max replay in both settings while remaining close in retention. Thus, a single RoD run achieves a strong adaptation–forgetting trade-off without tuning a replay ratio or training exhaustively on the available replay data.

Does RoD remain effective with proxy replay? For Qwen3.5, whose original pretraining data are unavailable, we instead use the Nemotron data as proxy replay. RoD continues to reach or improve upon the validation-loss frontier across both domains and model scales. On Legal, Qwen3.5-9B RoD



(a) **Forgetting mitigation.** Fixed replay largely preserves the dependence of remaining forgetting on knowledge category vulnerability, whereas RoD substantially flattens this relationship and provides stronger protection where forgetting is greatest.

(b) **Replay allocation.** Knowledge categories with greater forgetting under no-replay CPT are sampled more frequently relative to their prevalence in the replay pool, while well-retained categories are sampled less frequently ($r = 0.88$).

Figure 3: **RoD allocates replay according to forgetting demand.** On German adaptation with Nemotron-12B, RoD concentrates protection on the knowledge categories most vulnerable to forgetting (a) by allocating more replay to these categories during training (b).

matches the adaptation loss of 10% fixed replay (0.988) while reducing validation-loss forgetting from 0.075 to 0.031. On German, it achieves a comparable trade-off to 17% fixed replay (1.790 vs. 1.796 adaptation loss; 0.028 vs. 0.035 forgetting), with the same pattern at 4B. Higher replay shares further reduce forgetting at the cost of adaptation.

For task-based forgetting, RoD's advantage over fixed replay is smaller, which we attribute to the proxy replay loss being less well aligned with retention of Qwen's evaluated capabilities. We analyze this discrepancy in section C.2 and discuss the requirements on the replay distribution and signal in sections D.3 and D.4. Nevertheless, RoD substantially reduces task-based forgetting relative to no-replay CPT: by 60% on Legal Qwen3.5-9B and by 43% and 46% on German Qwen3.5-9B and 4B, respectively, at comparable adaptation.

Takeaways. RoD consistently reaches or improves upon the adaptation–forgetting frontier without choosing a replay ratio in advance. A single RoD run thereby reaches trade-offs that fixed-replay CPT obtains by explicitly sweeping the replay allocation.

4.2 RoD learns a demand-driven replay curriculum

The previous results show that RoD reaches a favorable adaptation–forgetting trade-off without specifying a replay ratio. We next examine the resulting curriculum: does RoD replay *what* is needed, *when* it is needed?

Does RoD replay what is being forgotten? We first ask whether RoD preferentially protects the parts of the general distribution most vulnerable to forgetting. Figure 3a relates forgetting under no-replay CPT to the forgetting remaining after replay for each knowledge category in the replay data. Fixed replay largely pushes down the no-replay forgetting profile: increasing the replay share reduces forgetting across categories, but those that forget more without replay continue to forget more after replay. In other words, fixed replay spends the same predetermined replay capacity irrespective of source-specific retention need, including on categories that would remain stable with substantially less replay. RoD instead rotates this profile toward the no-forgetting line, reducing its slope from 0.93 and 0.78 for fixed replay at 17% and 32% to 0.39 at an emergent replay share of 24%. Thus, RoD provides little additional protection where pretrained performance is already retained and concentrates its replay budget where adaptation causes substantial forgetting. We observe the same qualitative behavior across the remaining model–domain settings (section C.3).

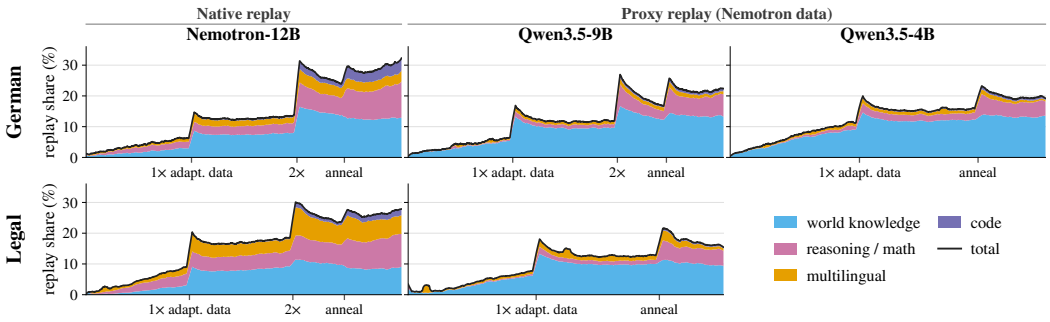


Figure 4: **RoD dynamically determines what to replay, how much, and when.** Stacked areas show the fraction of each training batch allocated to different knowledge categories, while the line shows the total replay share. Replay emerges as forgetting develops and is rebalanced throughout training, while its composition evolves across knowledge categories and model–domain settings. RoD thus adapts both the amount and composition of replay over training.

Figure 3b shows how this protection emerges from the replay allocation. It relates each category's vulnerability to its sampling ratio under RoD, with values above 1 indicating that a category is sampled more frequently than its prevalence in the replay pool. Categories with greater forgetting are systematically sampled more frequently relative to their share in the replay pool, while well-retained categories are sampled less frequently. For German adaptation of Nemotron-12B, this relationship is strong ($r = 0.88$): the most vulnerable web-scale knowledge categories are sampled at up to $2.1\times$ their prevalence in the replay pool, whereas well-retained categories such as math textbooks are sampled less frequently than their share. We observe the same positive relationship across all remaining model–domain settings (section C.3). Together, these results show that RoD allocates its replay budget according to what the model is forgetting, rather than simply reproducing the composition of the replay pool.

When and how much does RoD replay? Beyond deciding *what* to replay, RoD continuously adjusts the replay allocation as training progresses. Figure 4 shows that replay is initially absent because the target and base models coincide, giving replay samples zero reducible loss by construction. Early training therefore focuses predominantly on adaptation, while replay gradually increases as the target distribution is learned and forgetting emerges. This rebalancing is particularly evident around the $1\times$ and $2\times$ markers, which denote cumulative adaptation-data exposure equivalent to one and two dataset sizes.[1] As training progresses, the remaining learning potential of adaptation samples decreases on average, making replay increasingly competitive under joint selection. RoD consequently shifts toward replay as further adaptation becomes less valuable, reaching replay shares of 19–32% toward the end of training.

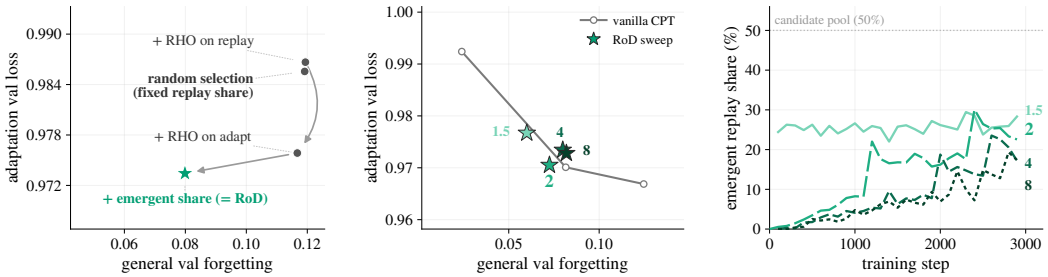
RoD simultaneously adapts *what* is replayed. The composition within the replay share changes throughout training and differs substantially across model–domain settings. For example, world knowledge accounts for most replay throughout the Qwen runs, whereas reasoning, mathematics, and multilingual data receive substantially larger shares for Nemotron, particularly later in training. These allocations themselves also evolve over time rather than remaining proportional to the replay pool, providing a temporal view of the source-specific replay allocation observed above.

Takeaways. RoD continuously balances adaptation against retention and retention demands within the replay distribution, determining *what* to replay, *how much*, and *when* based on the model's evolving learning and forgetting dynamics.

4.3 Joint adaptation–replay competition drives RoD

Having established how RoD's replay curriculum emerges, we next ask whether its gains arise from RHO-based data selection within a fixed replay allocation or from allowing the allocation itself to

[1]Training horizons differ across settings because we match them to the convergence of the corresponding no-replay run. Consequently, not all runs reach $2\times$ adaptation-data exposure.



(a) **Component ablation.** Joint competition improves upon selection at fixed replay allocation.

(b) **Candidate pool size.** The trade-off stabilizes once $m \geq 2$ leaves replay unconstrained.

(c) **Emergent replay share.** Replay trajectories stabilize for $m \geq 2$ under unconstrained allocation.

Figure 5: **Joint adaptation–replay competition drives RoD's gains.** Joint competition improves over fixed-allocation selection, while trade-off and replay allocation stabilize when unconstrained.

emerge. To isolate these effects, fig. 5a constructs a sequence of controlled fixed-allocation baselines that progressively introduce RoD's selection mechanisms. Starting from random selection, we first select replay examples by their RHO score and then apply RHO-based selection to both replay and adaptation examples, while keeping the replay share fixed. These baselines capture increasingly adaptive selection within a predefined replay allocation. RoD additionally removes this constraint and lets adaptation and replay examples compete jointly for the shared training budget.

Within a fixed replay budget, RHO-based replay selection largely maintains the adaptation–forgetting trade-off of random selection, while selecting adaptation examples improves adaptation. Allowing adaptation and replay examples to compete jointly further improves adaptation and substantially reduces forgetting, despite the same average replay share. Thus, RoD's gains cannot be explained by RHO-based selection within either stream alone. Instead, the gains emerge when adaptation and replay compete based on their respective reducible losses, determining what to replay and how much is needed.

We further examine RoD's candidate multiplier m, the primary selection hyperparameter in our implementation, which controls the number of candidates competing for each training batch. Under our balanced candidate construction, $m < 2$ implicitly imposes a minimum replay share as there are not enough adaptation candidates to fill the batch. At $m = 1.5$, this constraint leads to consistently high replay, resulting in low forgetting but weaker adaptation (fig. 5b–c). Once $m \geq 2$, adaptation candidates alone can fill the batch, and the replay allocation is determined entirely by joint selection. Increasing the candidate pool further provides additional selection freedom, but both the adaptation–forgetting trade-off and the resulting replay curriculum quickly stabilize, with similar behavior for $m = 2$, 4, and 8. We therefore use $m = 2$ throughout our main experiments, the smallest multiplier that leaves the replay allocation unconstrained.

Takeaways. Fixed-allocation baselines show that adaptive selection explains only part of RoD's gains. Joint adaptation–replay competition yields the strongest trade-off by adapting both replay content and allocation to the model's state. Once this replay allocation is unconstrained, the trade-off remains stable across candidate pool sizes.

4.4 RoD's model-dependent components transfer across model scales

RoD relies on model-dependent signals for both its adaptation reference and online data selection. Prior work provides evidence that data-selection signals can generalize across model scales (Brandfonbrener et al., 2024; Khaddaj et al., 2025). We therefore test whether RoD's adaptation reference and learned data curriculum similarly transfer across scales, providing a route to perform its model-dependent computation on smaller models.

Can the adaptation reference be smaller? We run RoD on Qwen3.5-9B using the corresponding Qwen3.5-4B model as its adaptation reference. As shown in table 1, the smaller reference recovers the adaptation–forgetting trade-off of native RoD across adaptation loss, validation-loss forgetting,

and task-based forgetting. Thus, the adaptation reference can operate below the target-model scale while recovering the resulting trade-off in this setting.

Can the curriculum be constructed at smaller scale? RoD's online selection induces a data curriculum by determining which adaptation and replay samples enter each batch. We construct this curriculum with a smaller model, record the selected batches, and use the resulting sequence to train a larger model without RoD selection at scale (table 1).

We first isolate this setting within Qwen3.5, where native 9B RoD provides a direct reference. Training Qwen3.5-9B on the curriculum constructed by Qwen3.5-4B recovers the adaptation–forgetting trade-off of native RoD despite a $2.25\times$ difference in model scale. The curriculum induced at 4B therefore captures learning and forgetting dynamics that remain informative for the 9B model.

Table 1: **Cross-scale RoD.** Larger target models use native RoD or components constructed by a smaller model (*ref.*: adaptation reference; *curr.*: curriculum). No-replay and fixed replay provide reference points.

Method	Adapt. ↓	Val. forget. ↓	Task forget. ↓
Qwen3.5-9B target *(4B → 9B;* $2.25\times$*)*			
No-replay	1.793	0.419	0.108
Fixed replay (32%)	1.807	0.019	0.039
RoD (native)	1.790	0.028	0.061
RoD (4B ref.)	1.796	0.013	0.047
RoD (4B curr.)	1.795	0.019	0.052
Nemotron-30B target *(12B → 30B;* $2.5\times$*)*			
No-replay	1.812	0.437	0.050
Fixed replay (32%)	1.818	0.130	0.023
RoD (12B curr.)	1.805	0.076	0.027
Qwen3.5-35B target *(4B → 35B;* $8.75\times$*)*			
No-replay	1.777	0.483	0.091
Fixed replay (32%)	1.753	0.027	0.044
RoD (4B curr.)	1.751	0.024	0.050

We then increase both the target-model scale and the relative scale gap. We train Nemotron-30B on a curriculum constructed by Nemotron-12B ($2.5\times$) and Qwen3.5-35B on a curriculum constructed by Qwen3.5-4B ($8.75\times$). In both settings, the smaller-model curriculum retains the adaptation performance of no-replay CPT while substantially reducing forgetting. At the same time, it achieves an adaptation–retention trade-off competitive with the fixed 32% replay baseline without specifying a replay ratio. Overall, RoD's learned data curriculum generalizes across model scales, remaining effective even when the target model is $8.75\times$ larger than the model used to construct it.

Takeaways. RoD's model-dependent components transfer across model scales. A smaller adaptation reference recovers native RoD at 9B, while curricula constructed by smaller models retain competitive adaptation–forgetting trade-offs for target models up to 35B and scale gaps of up to $8.75\times$. This allows the model-dependent computation underlying RoD to be performed at substantially smaller scale than the target model.

5 Conclusion

Continual pretraining must balance adapting to a new distribution with retaining capabilities acquired during pretraining. Replay can mitigate forgetting, but typically requires deciding in advance how much and what to replay, before the model's actual retention needs are known. We introduce **RoD**, which instead allows adaptation and replay examples to compete directly via complementary reducible-loss signals. Across domains, model families, and scales, RoD reaches or improves upon the adaptation–forgetting frontier of fixed replay without requiring a predefined replay ratio.

Our analyses show that RoD adapts replay to the model's evolving retention needs, targeting vulnerable sources and increasing replay as forgetting emerges. Ablations attribute these gains primarily to joint adaptation–replay competition rather than selection within either stream alone. Moreover, RoD's model-dependent computation can operate below the target-model scale: smaller adaptation references recover the trade-off of standard RoD, while transferred curricula remain effective across an $8.75\times$ scale gap.

Together, these results show that a strong stability–plasticity trade-off can emerge by replaying what is needed, when it is needed, rather than prescribing replay in advance. Our cross-scale results further provide practical paths for applying this principle to larger models.

AI USE STATEMENT

We used generative AI tools to assist with code implementation, experimental verification, and the integration of author-developed mathematical material into the manuscript. Specifically, LLMs were used to implement individual functions following author-provided method specifications, improve code efficiency, check experimental configurations for consistency, sanity-check experiments and their outputs, and translate author-provided mathematical derivations and notes into manuscript-ready form. We did not use generative AI tools to formulate hypotheses, develop theoretical or conceptual frameworks, independently derive or prove mathematical claims, design the research methodology or experiments, or independently interpret results; the remaining required-disclosure tasks are not applicable to this work.

Additionally, we used generative AI tools to refine scientific figures based on author-specified changes and to improve the language, clarity, and consistency of author-written manuscript drafts. All AI-assisted code, experimental checks, mathematical content, figures, and text were reviewed and verified by the authors. We take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies methods for continued pretraining of language models and does not involve human subjects or the collection of personal or sensitive data. Our experiments use existing language models and datasets for research purposes. We do not identify ethical concerns specific to the proposed methodology beyond those generally associated with the development and adaptation of large language models. We have conducted this work in accordance with the ICLR Code of Ethics.

REPRODUCIBILITY STATEMENT

We facilitate reproducibility by building all experiments on publicly available datasets and open-weight language models. We describe RoD and its training procedure in section 3, with the complete algorithm and implementation details provided in section A. The experimental setup is summarized in section 4 and documented in detail in section B, including the models and datasets, preprocessing, optimization and training configuration, baselines, evaluation procedure, compute setup, and random seed. We additionally provide an anonymized code repository containing the implementation of RoD and the code required to reproduce the experiments and evaluations reported in this work: `https://anonymous.4open.science/r/replay-on-demand-7042/README.md`.

REFERENCES

Nemo rl: A scalable and efficient post-training library. `https://github.com/NVIDIA-NeMo/RL`, 2025. GitHub repository.

Istabrak Abbes, Gopeshh Subbaraj, Matthew Riemer, Nizar Islah, Tsuguchika Tabaru, Hiroaki Kingetsu, Sarath Chandar, and Irina Rish. Revisiting replay and gradient alignment for continual pre-training of large language models. In Sarath Chandar, Razvan Pascanu, Eric Eaton, Bing Liu, Rupam Mahmood, and Amal Rannen-Triki (eds.), *Proceedings of The 4th Conference on Lifelong Learning Agents*, volume 330 of *Proceedings of Machine Learning Research*, pp. 465–486. PMLR, 11–14 Aug 2026. URL `https://proceedings.mlr.press/v330/abbes26a.html`.

Alankar Atreya, Devesh Batra, Yoages Kumar Mantri, Geremy Bantug, Greig A Cowan, and Raad Khraishi. When to review: Spaced repetition for continual pre-training of language models, 2026. URL `https://arxiv.org/abs/2608.17530`.

David Brandfonbrener, Hanlin Zhang, Andreas Kirsch, Jonathan Richard Schwarz, and Sham Kakade. CoLoR-Filter: Conditional loss reduction filtering for targeted language model pre-training. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3097. URL `https://proceedings.neurips.cc/paper_files/paper/2024/hash/b0f25f0a63cc544d506e4c1374a3c807-Abstract-Conference.html`.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: A strong, simple baseline. In *Advances in Neural Information Processing Systems*, 2020.

Jie Chen, Zhipeng Chen, Jiapeng Wang, Kun Zhou, Yutao Zhu, Jinhao Jiang, Yingqian Min, Wayne Xin Zhao, Zhicheng Dou, Jiaxin Mao, Yankai Lin, Ruihua Song, Jun Xu, Xu Chen, Rui Yan, Zhewei Wei, Di Hu, Wenbing Huang, and Ji-Rong Wen. Towards effective and efficient continual pre-training of large language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5779–5795, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.289. URL https://aclanthology.org/2025.acl-long.289/.

Sebastian Farquhar and Yarin Gal. A unifying bayesian view of continual learning. *arXiv preprint arXiv:1902.06494*, 2019.

Yujie Feng, Hao Wang, Jian Li, Xu Chu, Zhaolu Kang, Yiran Liu, Yasha Wang, Philip S. Yu, and Xiao-Ming Wu. FOREVER: Forgetting curve-inspired memory replay for language model continual learning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 24945–24965, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1144. URL https://aclanthology.org/2026.acl-long.1144/.

Jiawei Gu, Zacc Yang, Chuanghao Ding, Rui Zhao, and Fei Tan. CMR scaling law: Predicting critical mixture ratios for continual pre-training of language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 16143–16162, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.903. URL https://aclanthology.org/2024.emnlp-main.903/.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8342–8360, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.740. URL https://aclanthology.org/2020.acl-main.740/.

Adam Ibrahim, Benjamin Thérien, Kshitij Gupta, Mats L. Richter, Quentin Anthony, Timothée Lesort, Eugene Belilovsky, and Irina Rish. Simple and scalable strategies to continually pre-train large language models, 2024. URL https://arxiv.org/abs/2403.08763.

Xisen Jin, Dejiao Zhang, Henghui Zhu, Wei Xiao, Shang-Wen Li, Xiaokai Wei, Andrew Arnold, and Xiang Ren. Lifelong pretraining: Continually adapting language models to emerging corpora. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4764–4780, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.351. URL https://aclanthology.org/2022.naacl-main.351/.

Alaa Khaddaj, Logan Engstrom, and Aleksander Madry. Small-to-large generalization: Training data influences models consistently across scale. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=79ZkWgY2FI.

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.

Jeffrey Li, Mohammadreza Armandpour, Seyed Iman Mirzadeh, Sachin Mehta, Vaishaal Shankar, Raviteja Vemulapalli, Samy Bengio, Oncel Tuzel, Mehrdad Farajtabar, Hadi Pouransari, and Fartash Faghri. TiC-LM: A web-scale benchmark for time-continual LLM pretraining. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32231–32273, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1551. URL https://aclanthology.org/2025.acl-long.1551/.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, et al. Rho-1: Not all tokens are what you need. *arXiv preprint arXiv:2404.07965*, 2024.

Zheheng Luo, Xin Zhang, Xiao Liu, Haoling Li, Yeyun Gong, Qi Chen, and Peng Cheng. Velocitune: A velocity-based dynamic domain reweighting method for continual pre-training. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16644–16656, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.813. URL https://aclanthology.org/2025.acl-long.813/.

Sören Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, and Yarin Gal. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15630–15649. PMLR, 2022. URL https://proceedings.mlr.press/v162/mindermann22a.html.

NVIDIA. Nvidia nemotron nano 2: An accurate and efficient hybrid mamba-transformer reasoning model, 2025. URL https://arxiv.org/abs/2508.14444.

NVIDIA. Nemotron 3 ultra: Open, efficient mixture-of-experts hybrid mamba-transformer model for agentic reasoning, 2026. URL https://arxiv.org/abs/2606.15007.

Jupinder Parmar, Sanjev Satheesh, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Reuse, don't retrain: A recipe for continued pretraining of language models, 2024. URL https://arxiv.org/abs/2407.07263.

Haoran Que, Jiaheng Liu, Ge Zhang, Chenchen Zhang, Xingwei Qu, Yinghao Ma, Feiyu Duan, Zhiqi Bai, Jiakai Wang, Yuanxing Zhang, Xu Tan, Jie Fu, Wenbo Su, Jiamang Wang, Lin Qu, and Bo Zheng. D-cpt law: Domain-specific continual pre-training scaling law for large language models. *Advances in Neural Information Processing Systems*, 37:90318–90354, 2024.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Paper.pdf.

Karsten Roth, Vishaal Udandarao, Sebastian Dziadzio, Ameya Prabhu, Mehdi Cherti, Oriol Vinyals, Olivier Hénaff, Samuel Albanie, Matthias Bethge, and Zeynep Akata. A practitioner's guide to continual multimodal pretraining, 2024. URL https://arxiv.org/abs/2408.14471.

Jonathan Schwarz, Jelena Luketina, Wojciech M. Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress compress: A scalable framework for continual learning, 2018. URL https://arxiv.org/abs/1805.06370.

James Seale Smith, Lazar Valkov, Shaunak Halbe, Vyshnavi Gutta, Rogerio Feris, Zsolt Kira, and Leonid Karlinsky. Adaptive memory replay for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 3605–3615, June 2024. doi: 10.1109/CVPRW63382.2024.00364.

Soofi-Team, :, Benedikt Droste, David Fitzek, Ruben Härle, Lukas Helff, Maximilian Idahl, Alex Jude, Abbas Goher Khan, Maurice Kraus, Timm Ruland, Richard Rutmann, Sebastian Sztwiertnia, Markus Frey, Daniil Gurgurov, Jan Pfister, Tom Röhr, Sebastian von Rohrscheidt, Jörg Bienert, Nicolas Flores-Herr, Simon Gottschalk, Andreas Hotho, Kristian Kersting, Joachim Köhler, Alexander Löser, Wolfgang Nejdl, Simon Ostermann, Jan Plogsties, Björn Plüster, Patrick Putzky, Mehdi Ali, Michael Fromm, and Max Lübbering. A sovereign, open-source foundation model for german and english, 2026. URL https://arxiv.org/abs/2607.09424.

Lukas Thede, Stefan Winzeck, Zeynep Akata, and Jonathan Richard Schwarz. Captrack: Multifaceted evaluation of forgetting in llm post-training, 2026. URL https://arxiv.org/abs/2603.06610.

Raghuveer Thirukovalluru, Nicholas Monath, Bhuwan Dhingra, and Sam Wiseman. Sequence reducible holdout loss for language model pretraining. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (eds.), *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 14705–14716, Torino, Italia, May 2024. ELRA and ICCL. URL https://aclanthology.org/2024.lrec-main.1281/.

Michalis K Titsias, Jonathan Schwarz, Alexander G de G Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with gaussian processes. *arXiv preprint arXiv:1901.11356*, 2019.

Ruilin Tong, Yuhang Liu, Javen Qinfeng Shi, and Dong Gong. Coreset selection via reducible loss in continual learning. In *International Conference on Learning Representations*, 2025. URL https://api.semanticscholar.org/CorpusID:278395522.

Yifan Wang, Binbinliu, Fengze Liu, Yuanfan Guo, Jiyao Deng, Xuecheng Wu, Weidong Zhou, Xiaohuan Zhou, and Taifeng Wang. TiKMiX: Efficient semi-dynamic data mixture via data influence for LLM pre-training. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5777–5793, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.261. URL https://aclanthology.org/2026.acl-long.261/.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 23965–23998. PMLR, 2022.

Kailai Yang, Xiao Liu, Lei Ji, Hao Li, Xiao Liang, Zhiwei Liu, Yeyun Gong, Peng Cheng, and Mao Yang. Data mixing agent: Learning to re-weight domains for continual pre-training. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9447–9473, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.427. URL https://aclanthology.org/2026.acl-long.427/.

Çağatay Yıldız, Nishaanth Kanna Ravichandran, Nitin Sharma, Matthias Bethge, and Beyza Ermis. Investigating continual pretraining in large language models: Insights and implications, 2025. URL https://arxiv.org/abs/2402.17400.

A Additional Method Details

We provide additional motivation for RoD's source-specific reducible-loss scores and joint selection procedure, followed by the complete training algorithm and implementation details.

A.1 Deriving Source-Specific Reducible Loss from RHO

RoD uses different reference models for adaptation and replay: an adaptation specialist θ^A_{ref} estimates what remains to be learned on the adaptation distribution, while the pretrained model θ_0 measures degradation on replay data. We motivate this construction from the original RHO objective.

Let

$$L(x \mid \mathcal{S}) = -\frac{1}{n} \log p(x \mid \mathcal{S}) \tag{5}$$

denote the per-token loss under the posterior predictive of a Bayesian learner that has observed data $\mathcal{S}$, where $n = 4096$ is the number of loss-bearing tokens per candidate. During CPT, the learner has observed

$$\mathcal{S}_t = \mathcal{D}_0 \cup B_{1:t}, \tag{6}$$

consisting of the pretraining corpus $\mathcal{D}_0$ underlying θ_0 and the CPT batches $B_{1:t}$ encountered so far.

Following the derivation of RHO (Mindermann et al., 2022), consider a holdout $\mathcal{H} = \mathcal{H}_A \cup \mathcal{H}_R$ containing data from both the adaptation and replay distributions. The utility of a candidate x can be expressed through the improvement it induces in the likelihood of this holdout. By Bayes' rule,

$$\log p(\mathcal{H} \mid \mathcal{S}_t, x) - \log p(\mathcal{H} \mid \mathcal{S}_t) = n\big[L(x \mid \mathcal{S}_t) - L(x \mid \mathcal{S}_t, \mathcal{H})\big]. \tag{7}$$

Thus, candidate utility is determined by the difference between its loss under the current learner and the loss attainable after conditioning on the holdout.

To obtain the RoD scores used in practice, we make three approximations. First, as in RHO, we approximate the current posterior predictive with the current model,

$$L(x \mid \mathcal{S}_t) \approx \ell_{\theta_t}(x). \tag{8}$$

Second, we approximate the posterior conditioned on the holdout by dropping the particular CPT trajectory while retaining the original pretraining data,

$$L(x \mid \mathcal{S}_t, \mathcal{H}) \approx L(x \mid \mathcal{D}_0, \mathcal{H}). \tag{9}$$

Finally, we assume that the relevant component of the holdout depends on the source of the candidate:

$$L(x \mid \mathcal{D}_0, \mathcal{H}_A, \mathcal{H}_R) \approx \begin{cases} L(x \mid \mathcal{D}_0, \mathcal{H}_A) \approx \ell_{\theta^A_{\text{ref}}}(x), & x \in C_A, \\ L(x \mid \mathcal{D}_0) \approx \ell_{\theta_0}(x), & x \in C_R. \end{cases} \tag{10}$$

For replay examples, $\mathcal{D}_0$ already contains substantially more data from the replay distribution than the additional holdout $\mathcal{H}_R$, while adaptation data provide comparatively little information about $\mathcal{D}_R$. The pretrained model therefore provides the natural irreducible-loss reference for replay. For adaptation examples, θ^A_{ref} instead approximates the model obtained after learning the adaptation distribution. Importantly, its forgetting on the pretraining distribution does not enter the score because the adaptation reference is evaluated only on $\mathcal{D}_A$.

Together, these approximations recover the source-specific scores used by RoD,

$$\rho(x; \theta_t) = \begin{cases} \rho_A(x; \theta_t) = \ell_{\theta_t}(x) - \ell_{\theta^A_{\text{ref}}}(x), & x \in C_A, \\ \rho_R(x; \theta_t) = \ell_{\theta_t}(x) - \ell_{\theta_0}(x), & x \in C_R. \end{cases} \tag{11}$$

This derivation provides a common interpretation of both scores as excess loss relative to a source-appropriate reference.

Interpretation under exact Bayesian updating. Under exact Bayesian inference, sequential updating recovers the joint posterior (Farquhar & Gal, 2019). For replay examples, the dominance of the original pretraining data, $|\mathcal{D}_0| \gg |B_{1:t}|$, then implies

$$L(x \mid \mathcal{S}_t) \approx L(x \mid \mathcal{D}_0) \approx L(x \mid \mathcal{S}_t, \mathcal{H}), \qquad x \sim \mathcal{D}_R. \tag{12}$$

The corresponding ideal replay score is therefore approximately zero. Under the approximation $L(x \mid \mathcal{S}_t) \approx \ell_{\theta_t}(x)$, the practical replay score ρ_R can consequently be interpreted as measuring the per-example deviation of the continually trained model from this idealized sequential learner.

Why use source-specific references? Applying standard RHO directly would require a single reference trained on $\mathcal{H}_A \cup \mathcal{H}_R$. In CPT, such a model would itself correspond to one particular adaptation–retention trade-off, determined by the mixture of adaptation and replay data used to construct it. The desired posterior conditioned on both $\mathcal{D}_0$ and adaptation data therefore cannot generally be represented by a single SGD-trained reference without first choosing how to balance these objectives.

RoD instead approximates this reference separately for the two candidate sources. When the source-matched reference is the better of the two references for its corresponding examples, the resulting score coincides, up to at most $(\log 2)/n \approx 1.7 \times 10^{-4}$ nats per token, with RHO using an equal-weight Bayesian model average of the two references:

$$\rho(x) \approx \ell_{\theta_t}(x) - \min\{\ell_{\theta^A_{\text{ref}}}(x), \ell_{\theta_0}(x)\}. \tag{13}$$

This provides an alternative interpretation in which the two source-specific references approximate a common reference model while avoiding the need to solve the adaptation–retention trade-off during reference construction.

Why replay requires online scoring. The distinction between adaptation and replay also explains why the replay signal must depend on the current model. Replacing θ_t with θ_0 in the adaptation score gives

$$\ell_{\theta_0}(x) - \ell_{\theta^A_{\text{ref}}}(x), \tag{14}$$

which recovers the offline conditional loss-reduction signal used by CoLoR-Filter (Brandfonbrener et al., 2024). Applying the same substitution to replay instead gives

$$\ell_{\theta_0}(x) - \ell_{\theta_0}(x) = 0. \tag{15}$$

Unlike adaptation potential, forgetting is therefore not a static property of an example. It arises along the model's training trajectory and must be measured online.

A.2 Joint Selection as Resource Allocation

The previous derivation motivates the source-specific scores. We next provide an interpretation of why comparing these scores through joint top-k selection yields an adaptive allocation between adaptation and replay.

Both ρ_A and ρ_R measure excess code length, in nats per token, relative to the corresponding source-specific reference. For a relative retention weight $\kappa \geq 0$, consider

$$\begin{aligned} J_\kappa(\theta) &= \mathbb{E}_{x\sim\mathcal{D}_A}\big[\rho_A(x;\theta)\big] + \kappa\, \mathbb{E}_{x\sim\mathcal{D}_R}\big[\rho_R(x;\theta)\big] \\ &= \tfrac{1}{n}\big[D_{\mathrm{KL}}(\mathcal{D}_A\|p_\theta) - D_{\mathrm{KL}}(\mathcal{D}_A\|p_{\theta^A_{\text{ref}}})\big] + \tfrac{\kappa}{n}\big[D_{\mathrm{KL}}(\mathcal{D}_R\|p_\theta) - D_{\mathrm{KL}}(\mathcal{D}_R\|p_{\theta_0})\big]. \end{aligned} \tag{16}$$

This objective makes explicit the exchange rate between reducing adaptation loss and preserving the replay distribution.

For native replay, where $\mathcal{D}_0 \sim \mathcal{D}_R$, and under a flat prior, the negative log-posterior of Bayesian CPT on $\mathcal{D}_0$ and the adaptation corpus equals N_A times the empirical J_κ up to a constant, with

$$\kappa = \frac{N_0}{N_A}, \tag{17}$$

where N_0 and N_A denote the respective token counts. For Nemotron, $N_0 \approx 2 \times 10^{13}$ (NVIDIA, 2025) and $N_A \approx 5 \times 10^9$ (section B.2), giving $\kappa \approx 4 \times 10^3$. A fixed replay mixture with replay share r instead corresponds at convergence to

$$\kappa = \frac{r}{1-r}. \tag{18}$$

Practical CPT therefore deliberately chooses an operating point that trades some retention of the full pretraining posterior for stronger adaptation.

The replay component also connects to parameter-space regularization. For native replay,

$$\nabla_\theta \mathbb{E}_{\mathcal{D}_R}[\ell_\theta]\big|_{\theta_0} \approx 0. \tag{19}$$

Letting $\Delta = \theta - \theta_0$ and using the Fisher approximation F_R to the Hessian, valid when $p_{\theta_0} \approx \mathcal{D}_R$, gives

$$\mathbb{E}_{\mathcal{D}_R}[\rho_R] \approx \frac{1}{2}\Delta^\top F_R \Delta. \tag{20}$$

This corresponds to the full-Fisher form of the EWC penalty (Kirkpatrick et al., 2017; Schwarz et al., 2018). RoD targets the same local notion of deviation from the pretrained model through data allocation rather than by adding an explicit parameter-space regularizer.

Marginal utility of a training example. We can further relate the reducible-loss score to the expected utility of assigning a training slot to an example. Suppose that, for examples with $\rho(x) > 0$, the loss satisfies a per-example Polyak–Łojasiewicz-type condition relative to the reference loss,

$$\|\nabla \ell_\theta(x)\|^2 \geq 2\mu_s \rho(x), \qquad s = s(x) \in \{A, R\}, \tag{21}$$

and that ℓ is β-smooth. A gradient step on x with $\eta \leq 1/\beta$ then decreases its loss by at least

$$\eta \mu_s \rho(x). \tag{22}$$

Treating this bound as tight, neglecting interactions between examples as in RHO batch selection, and assuming that the reduction transfers to the corresponding source distribution, the gain from assigning one batch slot to x can be approximated as

$$g(x) \approx \eta\, w_s \mu_s\, \rho(x), \tag{23}$$

where $w_A = 1$ and $w_R = \kappa$.

Under these assumptions, maximizing

$$\sum_{x \in B_t} g(x) \qquad \text{subject to} \qquad |B_t| = k \tag{24}$$

is separable across candidates. Selecting the top-k examples therefore maximizes this local surrogate exactly. For

$$\kappa = \frac{\mu_A}{\mu_R}, \tag{25}$$

the source-dependent factors cancel and the selection reduces to the unweighted RoD rule,

$$B_t = \mathrm{TopK}_{x \in C_A \cup C_R}\, \rho(x; \theta_t). \tag{26}$$

This view provides a resource-allocation interpretation of RoD. The k-th largest score τ_t acts as the current shadow price of a training slot: examples from each source are selected until their marginal reducible loss falls below this threshold. The replay share is therefore not prescribed independently, but emerges from the relative distributions of adaptation and replay scores at each training step. RoD uses the unweighted comparison as a parameter-free default, placing one nat of adaptation reducible loss and one nat of replay reducible loss on equal footing.

A.3 Interpretation under Proxy Replay

The preceding interpretation is cleanest when replay examples originate from the model's own pretraining distribution. When only proxy replay data are available, as in our Qwen3.5 experiments, the pretrained model need not be converged on the replay distribution. This changes the interpretation of the replay score.

Let q denote the proxy replay distribution. If

$$\bar{g}_q = \nabla_\theta \mathbb{E}_q[\ell_\theta]\big|_{\theta_0} \neq 0, \tag{27}$$

and H_q denotes the corresponding Hessian, then

$$\mathbb{E}_{x\sim q}[\rho_R(x;\theta)] = \tfrac{1}{n}\big[D_{\mathrm{KL}}(q\|p_\theta) - D_{\mathrm{KL}}(q\|p_{\theta_0})\big] \approx \bar{g}_q^\top \Delta + \tfrac{1}{2}\Delta^\top H_q \Delta. \tag{28}$$

Unlike the native-replay case, the linear term does not vanish. The model can therefore reduce its replay score partly by fitting the proxy distribution q, rather than exclusively by returning toward the pretrained model θ_0. This provides one explanation for why validation-loss retention on proxy replay can be less tightly coupled to task-based retention, as observed for Qwen3.5 in section C.2.

One alternative would be to measure functional deviation from the pretrained model directly through self-distillation,

$$\rho_R^{\mathrm{KD}}(x) = \frac{1}{n}\sum_j D_{\mathrm{KL}}\big(p_{\theta_0}(\cdot \mid x_{<j}) \,\|\, p_{\theta_t}(\cdot \mid x_{<j})\big) \geq 0. \tag{29}$$

This score is minimized at θ_0 for any replay distribution q and would therefore remove the linear proxy-fitting term, in the spirit of distillation-based replay and functional regularization (Rolnick et al., 2019; Buzzega et al., 2020; Titsias et al., 2019). We do not use this variant in our experiments because it requires storing or recomputing the pretrained model's token-level predictive distributions on replay data, whereas our loss-based reference scores can be precomputed as a single scalar per sequence.

A.4 RoD Algorithm and Implementation

We provide the complete RoD training procedure in algorithm 1 and detail the candidate construction, loss computation, and reference-loss caching used in our implementation.

Candidate construction. Before training, both the adaptation corpus $\mathcal{D}_A$ and replay corpus $\mathcal{D}_R$ are tokenized and packed into fixed-length blocks of 4096 tokens. Documents are packed only within the same data source, with an end-of-document token inserted between concatenated documents. Each resulting block therefore belongs unambiguously to either the adaptation or replay distribution.

At each training step, RoD constructs a candidate pool of mk sequences, where k is the trained batch size and m the candidate multiplier. We balance this pool between adaptation and replay by subsampling the replay corpus once before training such that both sources occur with equal probability in the shuffled candidate stream. Consequently, $|C_A| = |C_R| = mk/2$ in expectation, independent of the relative sizes of the original corpora. Under this construction, $m = 2$ is the smallest multiplier for which adaptation candidates alone can fill the complete training batch; for $m < 2$, a minimum replay share is imposed by construction. We therefore use $m = 2$ by default and study this choice in section 4.3.

Sequence-level loss and reference caching. We score each candidate using its token-normalized, full-sequence language-modeling loss

$$\ell_\theta(x) = \frac{1}{|T(x)|}\sum_{j\in T(x)} -\log p_\theta(x_j \mid x_{<j}), \tag{30}$$

where $T(x)$ denotes the loss-bearing token positions in sequence x. All tokens in our packed pretraining sequences are loss-bearing. The same aggregation is used for the target and reference models, ensuring that their loss differences are directly comparable.

Because both reference models remain fixed throughout training, their losses are precomputed once. The adaptation specialist θ_{ref}^A scores every adaptation block, while the pretrained model θ_0 scores every replay block. Each loss is stored together with a deterministic hash of the block's source and token IDs. During training, candidate sequences are matched to their cached reference losses through this hash; neither reference model therefore requires an online forward pass.

Online scoring and joint selection. At each step t, the current model θ_t first performs a forward-only pass over the complete candidate pool $C_A \cup C_R$. We combine the resulting current-model losses with the cached reference losses to compute

$$\rho(x;\theta_t) = \begin{cases} \ell_{\theta_t}(x) - \ell_{\theta_{\mathrm{ref}}^A}(x), & x \in C_A, \\ \ell_{\theta_t}(x) - \ell_{\theta_0}(x), & x \in C_R. \end{cases} \tag{31}$$

Algorithm 1 RoD: joint reducible-loss selection for continued pretraining.

Require: adaptation data $\mathcal{D}_A$, replay data $\mathcal{D}_R$, adaptation reference θ^A_{ref}, pretrained model θ_0, batch size k, candidate multiplier m

Initialize target model $\theta \leftarrow \theta_0$
Precompute $\ell_{\theta^A_{\text{ref}}}(x)$ for all $x \in \mathcal{D}_A$
Precompute $\ell_{\theta_0}(x)$ for all $x \in \mathcal{D}_R$
for each training step **do**
    Draw mk candidates from the balanced mixture of $\mathcal{D}_A$ and $\mathcal{D}_R$, yielding C_A and C_R
    Compute $\ell_\theta(x)$ for all $x \in C_A \cup C_R$ ▷ forward only
    for $x \in C_A \cup C_R$ **do**
        if $x \in C_A$ **then**
            $\rho(x) \leftarrow \ell_\theta(x) - \ell_{\theta^A_{\text{ref}}}(x)$
        else
            $\rho(x) \leftarrow \ell_\theta(x) - \ell_{\theta_0}(x)$
        end if
    end for
    $B \leftarrow \text{TopK}_{x \in C_A \cup C_R}\, \rho(x)$ ▷ $|B| = k$
    Update θ on B using the standard language-modeling loss
end for

RoD then jointly ranks all mk candidates and selects the k highest-scoring sequences,

$$B_t = \text{TopK}_{x \in C_A \cup C_R}\, \rho(x; \theta_t). \tag{32}$$

No source-specific quota is applied: adaptation and replay candidates compete directly for the same k training slots. Padding introduced for batch-size divisibility is assigned a score of $-\infty$ and cannot enter the selected batch.

Finally, we perform a standard forward–backward pass on B_t and update the model using the unweighted autoregressive language-modeling objective. RHO therefore affects training only through data selection; selected examples are not reweighted according to their scores. Each RoD step consequently consists of one forward-only scoring pass over mk candidates, a cached reference-loss lookup and joint top-k selection, followed by one forward–backward pass over the selected k examples. All reference-model computation is performed offline.

A.5 Adaptation Reference Construction

For each model–domain setting, we obtain the adaptation reference θ^A_{ref} by continuing pretraining of the corresponding base model exclusively on the adaptation corpus, without replay or other forgetting mitigation. We first train with a warmup followed by a constant learning rate until the adaptation validation loss converges, which occurs after approximately two epochs across our settings. Once convergence is reached, we append an additional 0.5 epochs of cosine learning-rate decay and use the resulting checkpoint as θ^A_{ref}. The same model constitutes the no-replay CPT baseline in our experiments.

These adaptation specialists achieve the strongest, or comparable-to-strongest, adaptation performance among our training runs, but exhibit substantial forgetting of the pretrained distribution. This forgetting is inconsequential for their role in RoD: the adaptation reference is not intended to provide a desirable adaptation–retention trade-off, but to estimate the loss attainable after learning the adaptation distribution. Its loss therefore provides a target against which RoD measures the remaining learning potential of each adaptation example. Moreover, this estimate need not be produced at the target-model scale: in section 4.4, we show that a 4B adaptation reference recovers the adaptation–forgetting trade-off of the standard 9B reference within the Qwen3.5 family.

B Extended Experimental Setup

We provide additional details on the models, data, training setup, baselines, and evaluation used in our experiments. Our experimental settings are designed to progressively test RoD beyond its

primary Nemotron-12B setting. We first evaluate Nemotron-12B on both Legal and German adaptation, where the original pretraining distribution is available for replay. We then evaluate Qwen3.5-9B on the same two domains to test whether RoD generalizes across model families and remains effective when the original pretraining data are unavailable and replay must instead rely on a proxy distribution. Qwen3.5-4B provides an additional smaller-scale setting within the Qwen family and, more importantly, serves as the source model for our cross-scale curriculum-transfer experiments; we therefore evaluate it only on German, the domain used for these scaling experiments. Finally, we test scalability to substantially larger models by transferring RoD curricula constructed at smaller scale to Nemotron-30B-A3B and Qwen3.5-35B-A3B, avoiding online curriculum construction at the target-model scale. Together, these settings separate generalization across domains and model families from scalability through curriculum transfer.

Unless stated otherwise, all methods within a model–domain setting use the same preprocessing, optimization setup, and trained-token budget. All experiments use a sequence length of 4096 tokens and a trained batch of 1024 sequences, corresponding to approximately 4.19M trained tokens per update.

B.1 Models

We use Nemotron-Nano-12B-v2 and Qwen3.5-9B for the primary Legal and German experiments, with Qwen3.5-4B additionally evaluated on German. For the cross-scale curriculum-transfer experiments in section 4.4, we use Nemotron-3-Nano-30B-A3B and Qwen3.5-35B-A3B as larger target models.

Nemotron-Nano-12B-v2 (NVIDIA, 2025) is a 12B-parameter hybrid architecture combining Mamba-style state-space layers with periodic attention. For the large-scale transfer experiment, we additionally use Nemotron-3-Nano-30B-A3B (NVIDIA, 2026), a 30B-parameter mixture-of-experts model with approximately 3B active parameters per token. Qwen3.5-9B and Qwen3.5-4B (Qwen Team, 2026) are dense hybrid models combining Gated DeltaNet layers with periodic full attention. We additionally use Qwen3.5-35B-A3B (Qwen Team, 2026) as the target of our largest curriculum-transfer experiment. Although the Qwen3.5 checkpoints use a multimodal model class, all continued-pretraining experiments are text-only. We use each model's corresponding tokenizer, with the 4B and 9B Qwen3.5 models sharing the same tokenizer.

B.2 Adaptation and Replay Data

Adaptation data. For Legal adaptation, we use the Nemotron-Pretraining-Legal-v1 corpus (NVIDIA, 2026). The corpus contains legal text from multiple sources, including case law, regulatory text, and legal question-answering data, and is dominated by CaseHold, which accounts for approximately 82% of the adaptation corpus. For German adaptation, we construct a German-language corpus following the data collection of Soofi-Team et al. (2026). The corpus comprises nine sources spanning German PDF and Wikipedia data as well as cultural, economic, legal, news, political, scientific, and web data.

Replay data. For replay, we use the general Nemotron pretraining corpus (NVIDIA, 2025). It contains data from 14 fine-grained sources spanning world knowledge, reasoning and mathematics, multilingual data, and code. For Nemotron, these data originate from its own pretraining distribution and therefore provide a direct approximation to the knowledge acquired during pretraining. The original Qwen3.5 pretraining data are not publicly available. We therefore use the same Nemotron general corpus as a proxy for replay in Qwen3.5. As discussed in sections C.2 and D.3, this distinction is important because loss degradation on proxy replay need not be as closely aligned with capability forgetting as degradation on the model's own pretraining distribution.

Preprocessing and splits. We tokenize all corpora using the respective target model's tokenizer and pack the documents into sequences of 4096 tokens. Packing is performed within each source so that every sequence retains a unique source label for our source-level analyses. Partial final sequences are discarded. We associate every packed sequence with a content hash, which allows us to consistently match examples between reference-loss precomputation, training, and evaluation. We construct a deterministic 1% validation holdout set independently for each source, using the content

hash and seed 42. These examples are excluded from training and form the source-specific validation sets used to measure adaptation and forgetting. For German, we subsample the adaptation corpus to 25% of its documents and cap individual sources to balance coverage across the nine constituent sources. Replay data are subsampled to 20% of the available documents for both domains. After preprocessing and subsampling, the resulting Legal and German corpora contain approximately 9.6B and 14.2B tokens, respectively. For Legal, adaptation and replay data form an approximately balanced 50:50 mixture, while the German corpus contains approximately 35% adaptation and 65% replay data. For RoD, we balance the candidate stream between adaptation and replay before joint selection. Consequently, the available corpus proportions do not impose a minimum replay share on RoD; the selected replay fraction is determined by the joint RHO ranking.

B.3 Training Setup

Optimization. We train all continued-pretraining models using Adam with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, weight decay 0.1, and gradient clipping at 1.0. Training uses bfloat16 precision and a distributed optimizer. We select the learning rate through preliminary adaptation experiments rather than tuning it separately for RoD. An initial learning-rate pilot at smaller batch scale identified 1×10^{-5} as the strongest setting among the evaluated values. We subsequently scale and tune the learning rate for the full training configuration, resulting in a peak learning rate of 1.2×10^{-4}, which we keep fixed across all primary model–domain settings. The minimum learning rate is 1.2×10^{-5}.

We use a warmup–stable–decay schedule. After a short warm-up, training proceeds at a constant learning rate until the adaptation validation loss of the corresponding no-replay baseline converges, followed by a cosine decay to the minimum learning rate. The resulting trained-token-matched budgets range from 2,824 to 2,980 updates across the primary model–domain settings, corresponding to approximately 11.8–12.5B trained tokens. Within each setting, RoD and all trained-token-matched fixed-replay baselines use the same optimization schedule and trained-token budget.

Batch construction. Each training update contains 1024 sequences of length 4096, corresponding to 4,194,304 trained tokens. For RoD, the candidate multiplier m controls the number of sequences considered for selection while the trained batch remains fixed. We use $m = 2$ throughout the main experiments and vary only m in the candidate-multiplier ablation (Section 4.3). The construction and selection of RoD candidates are described in Section A.

Cross-scale curriculum transfer. For the cross-scale experiments in section 4.4, we construct the RoD curriculum using a smaller source model and reuse it to train a larger target model without performing online selection at the target scale. Specifically, we record the training batches selected by RoD throughout the source-model run, preserving both their adaptation–replay composition and ordering, and train the target model on this fixed sequence of examples. The resulting curriculum therefore transfers RoD's learned allocation of what to train on and when, while avoiding the model-dependent reference-loss computation and candidate selection at the larger target scale.

Runs and compute. We use seed 42 throughout training and run each experimental condition once. Accordingly, the reported frontiers are single-seed estimates. We assess the consistency of our findings by replicating across model families, model scales, and adaptation domains, rather than by repeated runs of individual settings. Small differences between nearby operating points should therefore not be interpreted as statistically significant. Our conclusions instead rely on qualitative patterns that recur across model families, scales, adaptation domains, and replay settings. Training is distributed across NVIDIA B200 GPUs, with the number of GPUs adjusted to the model and experimental setting. Reference losses are computed separately before RoD training using forward-only evaluation and cached by content hash. Thus, the reference models introduce an offline computation cost but do not need to be evaluated during RoD training.

B.4 Baselines

We compare RoD against three strategies for navigating the adaptation–retention trade-off: continued pretraining without replay, continued pretraining with fixed replay allocations, and post-hoc model merging.

No-replay CPT. The no-replay baseline continues pretraining exclusively on the adaptation corpus without replay. It therefore represents the adaptation-focused endpoint of the replay frontier and exposes the forgetting that arises in the absence of explicit retention measures. The same training procedure is used to construct the domain specialist that serves as RoD's adaptation reference.

Fixed-replay CPT. Fixed-replay baselines mix adaptation and replay examples at a constant ratio throughout training without data selection. We construct three replay settings by retaining 0.11, 0.25, and 1.0 of the available replay corpus while keeping the adaptation corpus fixed. Because the available adaptation–replay corpus ratios differ between domains, these same subsampling configurations result in different realized training mixtures. For Legal, they correspond to approximately 10%, 20%, and 50% replay; for German, they correspond to approximately 17%, 32%, and 65% replay.

For our primary comparison, we train each fixed-replay baseline with the same trained-token budget as the no-replay CPT baseline. Sweeping the replay allocation provides an oracle-like comparison that approximates the operating points available when the replay ratio can be selected in hindsight, whereas RoD determines its replay allocation within a single run. We additionally report a *max-replay* reference trained on all available adaptation and replay data. These runs use approximately twice the trained-token budget and are therefore treated as higher-budget references rather than trained-token-matched baselines. Section C.1 further examines this distinction by continuing the German fixed-replay baselines beyond the matched trained-token budget until convergence.

Model merging. As a post-hoc alternative to replay, we linearly interpolate the parameters of the pretrained base model θ_0 and the no-replay model θ_A, following the weight-space interpolation used in model soups (Wortsman et al., 2022),

$$\theta_{\text{merge}} = (1 - \lambda)\theta_0 + \lambda\theta_A, \tag{33}$$

with $\lambda \in \{0.2, 0.4, 0.6, 0.8\}$. Merging is performed independently for every floating-point parameter and requires no additional training. Varying λ yields a post-hoc adaptation–retention trade-off between the pretrained and adapted models.

Across training-based methods, we keep the optimizer, learning-rate schedule, sequence length, trained batch size, seed, and matched trained-token budget fixed within each model–domain setting. The primary difference, therefore, lies in how the training budget is allocated between adaptation and replay.

B.5 Evaluation

We evaluate continual pretraining along two complementary dimensions: adaptation to the target distribution and retention of capabilities acquired during pretraining.

Validation-loss evaluation. We measure adaptation using language-modeling loss on held-out data from the adaptation corpus. Our objective is specifically to model the target distribution used for continued pretraining, rather than to optimize a predefined set of downstream Legal or German capabilities. Held-out language-modeling loss therefore directly measures how well the model has learned this distribution without introducing assumptions about which downstream tasks should benefit from adaptation. We compute loss separately for every adaptation source and aggregate sources according to their corpus prevalence, such that lower adaptation loss indicates stronger modeling of the target distribution.

We analogously evaluate language-modeling loss on held-out general data to measure forgetting at the distribution level. For each source o, we compare the continually pretrained model's loss $\ell_\theta(o)$ with that of the pretrained base model $\ell_{\theta_0}(o)$. We define validation-loss forgetting as

$$F_{\text{val}}(\theta) = \sum_{o \in \mathcal{D}_R} w_o \max(0,\, \ell_\theta(o) - \ell_{\theta_0}(o))\,, \tag{34}$$

where w_o denotes the source's corpus weight. Capping source-level differences at zero prevents improvements on one source from compensating for degradation on another. Both the base model and all continually pretrained checkpoints are evaluated on the same source-specific holdouts. We

Table 2: **Task-based capability evaluation.** Benchmarks used to evaluate the four pretraining-time capability groups considered in our experiments.

Capability	Evaluation tasks
Parametric knowledge	MMLU World Religions, Prehistory, High School Geography, High School Government and Politics, US Foreign Policy, Security Studies, Global Facts, Miscellaneous, Clinical Knowledge, Medical Genetics, Professional Medicine, Anatomy, Management, Marketing, Business Ethics, and Professional Accounting; SciQ; OpenBookQA; ARC-Easy; MedQA; MMLU-Pro.
Reasoning and problem solving	BBH; MuSR; MMLU Formal Logic, Logical Fallacies, Econometrics, High School Microeconomics, High School Macroeconomics, High School Mathematics, College Mathematics, Elementary Mathematics, Abstract Algebra, and High School Statistics; ARC-Challenge; AQuA-RAT; SAT Math.
Commonsense and robustness	HellaSwag; PIQA; WinoGrande; CommonsenseQA.
Multilingual capabilities	multilingual MMLU in German, French, Spanish, Russian, Chinese, and Arabic; BeleBele in German, French, Spanish, Russian, Chinese, Arabic, and English.

use the corpus-weighted metric throughout the main paper and additionally retain unweighted source averages as a breadth-oriented diagnostic.

This distribution-level metric is conceptually aligned with RoD's replay signal, as both quantify degradation in language-modeling loss relative to the pretrained model. Importantly, forgetting is evaluated on held-out data excluded from training, testing whether RoD's example-level replay selection translates into retention beyond the selected training examples.

Task-based capability evaluation. Validation loss measures retention directly on the replay distribution, but changes in language-modeling loss do not necessarily translate into changes in downstream capabilities. We therefore complement it with a task-based evaluation organized according to the capability taxonomy introduced by CapTrack (Thede et al., 2026). Because we evaluate base models without post-training, we focus on latent capabilities that can be meaningfully assessed in pretrained models. Specifically, we evaluate four capability groups: *parametric knowledge*, *reasoning and problem solving*, *commonsense and robustness*, and *multilingual capabilities*. Table 2 lists the benchmarks used to instantiate each capability.

All tasks are evaluated as multiple-choice problems using our evaluation suite with a vLLM backend. Following the underlying benchmark configurations, we report length-normalized accuracy for tasks that define it and raw accuracy otherwise. Base and continually pretrained models are evaluated using identical task configurations and prompting setups.

For each capability, we aggregate performance across its constituent tasks and compute the decrease relative to the corresponding base model, capped at zero. Task-based forgetting is then defined as the mean forgetting across the four capability groups. As for validation-loss forgetting, this construction prevents improvements in one capability from compensating for degradation in another.

C Extended Experiments and Analyses

C.1 Additional Adaptation–Forgetting Results

Full frontier results. Table 3 reports the complete numerical results underlying the adaptation–forgetting frontiers in fig. 2. We report adaptation loss, validation-loss forgetting, and task-based forgetting for all fixed-replay, model-merging, and RoD runs, together with their realized replay shares and training budgets. Except for the explicitly marked max-replay runs, all training-based methods within each setting are trained-token-matched.

Training fixed-replay baselines to convergence. Beyond the trained-token-matched comparison in the main paper, we additionally evaluate fixed-replay CPT, allowing each run to train until con-

Table 3: **Full adaptation–forgetting frontier, all settings.** Adaptation loss, general validation-loss forgetting, and task-based forgetting (capped mean accuracy drop over four capabilities) for every arm plotted in fig. 2. All arms within a setting are compute-matched (same optimizer steps from base) except *max-replay*, trained to its own epoch target at $\approx 2\times$ the tokens. Replay share is the realized fraction of the trained batch for fixed-replay arms and the emergent fraction for RoD; model merging has no replay share (weight-space interpolation). Val. forget. and task forget. are $\downarrow$ (0 = no forgetting); adapt. is $\downarrow$. Legal · Qwen3.5-9B has no off-budget max-replay arm (never trained).

Setting	Method	Replay share	Adapt. ↓	Val. forget. ↓	Task forget. ↓	Tokens (B)
Legal · Nemotron-12B	Base	–	1.492	0.000	0.000	0.00
	No-replay	0%	0.972	0.553	0.122	12.41
	Fixed replay (10%)	10%	0.967	0.125	0.067	12.41
	Fixed replay (20%)	20%	0.970	0.082	0.055	12.41
	Fixed replay (50%)	50%	0.992	0.024	0.036	12.41
	Max-replay (off-budget)	50%	0.976	0.057	0.046	24.31
	Model merging (0.2)	–	1.225	0.038	0.000	12.41
	Model merging (0.4)	–	1.100	0.135	0.005	12.41
	Model merging (0.6)	–	1.013	0.256	0.030	12.41
	Model merging (0.8)	–	0.970	0.390	0.064	12.41
	RoD	21%	**0.967**	**0.071**	**0.049**	12.41
German · Nemotron-12B	Base	–	2.478	0.000	0.000	0.00
	No-replay	0%	1.793	0.407	0.096	12.50
	Fixed replay (17%)	17%	1.810	0.122	0.067	12.50
	Fixed replay (32%)	32%	1.813	0.077	0.049	12.50
	Fixed replay (65%)	65%	1.872	0.051	0.029	12.50
	Max-replay (off-budget)	65%	1.808	0.051	0.039	24.80
	Model merging (0.2)	–	2.261	0.027	0.000	12.50
	Model merging (0.4)	–	2.076	0.095	0.005	12.50
	Model merging (0.6)	–	1.912	0.186	0.021	12.50
	Model merging (0.8)	–	1.815	0.289	0.052	12.50
	RoD	24%	**1.788**	**0.073**	**0.050**	12.50
German · Qwen3.5-9B	Base	–	2.164	0.000	0.000	0.00
	No-replay	0%	1.793	0.420	0.108	12.42
	Fixed replay (17%)	17%	1.796	0.035	0.052	12.42
	Fixed replay (32%)	32%	1.807	0.019	0.039	12.42
	Fixed replay (65%)	65%	1.867	0.007	0.029	12.42
	Max-replay (off-budget)	65%	1.813	0.005	0.030	24.62
	Model merging (0.2)	–	2.044	0.021	0.001	12.42
	Model merging (0.4)	–	1.938	0.091	0.006	12.42
	Model merging (0.6)	–	1.853	0.188	0.020	12.42
	Model merging (0.8)	–	1.809	0.292	0.051	12.42
	RoD	16%	**1.790**	**0.028**	**0.061**	12.42
German · Qwen3.5-4B	Base	–	2.322	0.000	0.000	0.00
	No-replay	0%	1.849	0.521	0.197	12.00
	Fixed replay (17%)	17%	1.863	0.036	0.090	12.00
	Fixed replay (32%)	32%	1.878	0.020	0.079	12.00
	Fixed replay (65%)	65%	1.919	0.006	0.067	12.00
	Max-replay (off-budget)	65%	1.865	0.003	0.056	27.21
	Model merging (0.2)	–	2.188	0.035	0.003	12.00
	Model merging (0.4)	–	2.060	0.140	0.019	12.00
	Model merging (0.6)	–	1.944	0.274	0.062	12.00
	Model merging (0.8)	–	1.876	0.374	0.122	12.00
	RoD	18%	**1.859**	**0.028**	**0.107**	12.00
Legal · Qwen3.5-9B	Base	–	1.398	0.000	0.000	0.00
	No-replay	0%	0.986	0.571	0.147	11.84
	Fixed replay (10%)	10%	0.988	0.075	0.070	11.84
	Fixed replay (20%)	20%	0.992	0.044	0.055	11.84
	Fixed replay (50%)	50%	1.015	0.016	0.037	11.84
	Model merging (0.2)	–	1.219	0.041	0.000	11.84
	Model merging (0.4)	–	1.107	0.139	0.003	11.84
	Model merging (0.6)	–	1.026	0.274	0.025	11.84
	Model merging (0.8)	–	0.986	0.412	0.076	11.84
	RoD	15%	**0.988**	**0.031**	**0.059**	11.84

vergence. As shown in Figure 6, additional training primarily improves adaptation, shifting the fixed-replay operating points downward. However, these gains require substantially larger training budgets, reaching up to $1.98\times$ for Nemotron-12B and $2.27\times$ for Qwen3.5-4B. Even with these additional training budgets, RoD at $1\times$ compute remains on or improves upon the resulting adaptation–

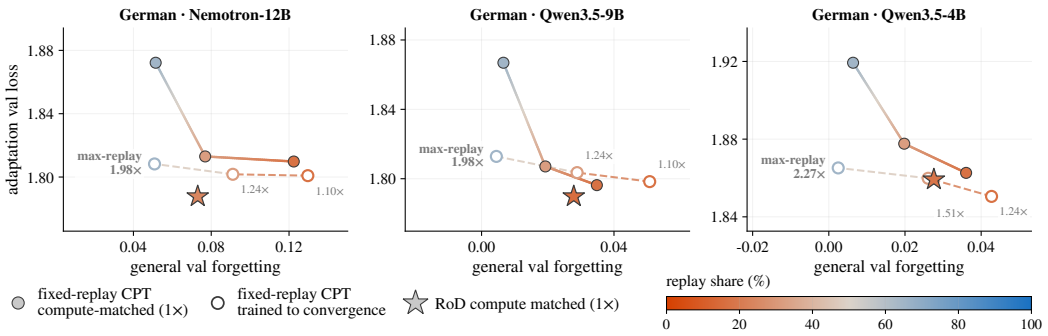


Figure 6: **Fixed-replay CPT trained to convergence.** Adaptation–forgetting trade-offs for German adaptation when fixed-replay CPT is trained beyond the matched trained-token (1×) budget until convergence. Filled circles show the trained-token-matched fixed-replay runs used in the main comparison, while open circles show the corresponding runs trained to convergence; annotations indicate their training budget relative to the trained-token-matched runs. RoD (star) is shown at the 1× compute budget. Extending fixed-replay training improves adaptation but does not recover a consistently better adaptation–forgetting trade-off than RoD, while requiring up to 2.27× the training budget.

Table 4: **RoD performance beyond the compute-matched training budget.** Continuing RoD beyond the 1× budget does not improve either adaptation or retention, indicating that RoD has already converged at the budget used in our main comparison.

Setting	Budget	Adapt. ↓	Val. forget. ↓
German · Nemotron-12B	1.00×	1.7878	0.0731
	1.19×	1.7882	0.0734
Legal · Nemotron-12B	1.00×	0.9670	0.0714
	1.20×	0.9705	0.0725

forgetting frontier. Thus, the favorable trade-off of RoD is not an artifact of restricting fixed replay to the trained-token-matched budget.

We additionally continue RoD for approximately 20% beyond the main training budget for German and Legal adaptation with Nemotron-12B. As shown in Table 4, neither adaptation nor retention improves with additional training, indicating that RoD has already converged at the 1× budget. Together, these results show that additional training can narrow the adaptation gap for fixed replay, but does not yield a better adaptation–forgetting frontier than RoD despite requiring substantially more compute.

C.2 Proxy Replay and Task-Based Forgetting

For Nemotron, RoD draws replay examples from the model's own pretraining corpus, such that the pretrained model provides a natural reference for measuring degradation on these data. For Qwen3.5, the original pretraining data are unavailable, and we instead use the Nemotron pretraining corpus as proxy replay. As discussed in Section 4.1, RoD continues to perform well in terms of validation-loss forgetting in this setting, but its advantage is less pronounced under task-based evaluation. We examine this discrepancy in more detail below.

Qwen is less well fit to the proxy replay distribution. A central difference between native and proxy replay is the pretrained model's fit to the replay distribution. Table 5 reports the base-model loss on the held-out replay data, both overall and across the four broad replay categories. The Nemotron base model achieves lower loss on these data than either Qwen3.5 model. Its overall replay loss ranges from 0.81 to 0.90 across our Legal and German setups, compared with 1.02–1.03 for Qwen3.5-9B and 1.08 for Qwen3.5-4B. This difference is consistent across world knowledge, reasoning and mathematics, multilingual data, and code.

Table 5: **Base-model loss on the replay distribution.** We report held-out language-modeling loss on the Nemotron replay corpus overall and across its four broad source categories. Nemotron is evaluated on replay data from its own pretraining distribution, whereas the same corpus serves as proxy replay for Qwen3.5. The higher Qwen losses indicate that the Qwen base models are less converged on the proxy replay distribution.

Setting	World knowl.	Reason./math	Multiling.	Code	Overall
Legal · Nemotron-12B	1.30	0.49	1.07	0.40	0.81
German · Nemotron-12B	1.36	0.49	1.08	0.43	0.90
Legal · Qwen3.5-9B	1.47	0.60	1.51	0.53	1.03
German · Qwen3.5-9B	1.46	0.60	1.51	0.53	1.02
German · Qwen3.5-4B	1.55	0.64	1.61	0.56	1.08

This distinction affects how the replay signal should be interpreted. RoD scores replay candidates relative to the pretrained model, $\rho_R(x;\theta_t) = \ell_{\theta_t}(x) - \ell_{\theta_0}(x)$. For native replay, the reference loss is measured on the model's own pretraining distribution and therefore directly captures degradation relative to its pretrained state on that distribution. For proxy replay, the same base-relative comparison remains a meaningful measure of degradation on the available replay data, but the data are less closely aligned with the distribution on which the model was originally trained. Consequently, validation-loss forgetting remains informative about retention on the proxy distribution, while its correspondence to changes in the model's original pretraining capabilities may be weaker. This motivates our complementary task-based evaluation, which directly measures whether the same trends extend to downstream capabilities.

The discrepancy is concentrated in reasoning-intensive capabilities. We next decompose task-based forgetting by capability to identify where proxy replay diverges from the validation-loss signal. Figure 7 shows the change in accuracy relative to the pretrained Qwen3.5-9B model for German adaptation. No-replay CPT degrades all four capability groups, with the largest drops in multilingual and robustness/commonsense evaluations. Both RoD and fixed replay substantially mitigate these losses, confirming that proxy replay still provides a useful retention signal.

The remaining difference between RoD and fixed replay is not uniform across capabilities. Parametric knowledge is retained comparatively well by RoD, closely matching the fixed-replay baselines. The clearest gap instead occurs for reasoning and mathematics: RoD reduces the no-replay accuracy drop from approximately 10 points to 6 points, but retains less performance than either fixed-replay baseline. A similar pattern appears in the multilingual group, where RoD improves substantially over no-replay but still lags behind fixed replay. This connection is consistent with the composition of our multilingual evaluations, which contain a substantial fraction of translated mathematical and reasoning tasks. Thus, part of the apparent multilingual forgetting reflects the same reasoning-intensive capabilities for which proxy replay provides weaker protection.

Together, these results clarify the discrepancy between the validation loss and the task-based forgetting observed for Qwen3.5. Proxy replay remains effective; RoD substantially reduces task-based forgetting relative to no-replay CPT, but the replay distribution is less closely matched to the knowledge acquired during Qwen's original pretraining. As a result, base-relative loss changes on the proxy corpus provide a noisier signal of capability retention than under native replay. This limitation is most visible for reasoning-intensive capabilities, where strong validation-loss retention does not translate as directly into task-level retention. We therefore view the Qwen experiments as a more challenging proxy-replay setting rather than a direct test of RoD with access to the model's original pretraining distribution.

C.3 Demand-Driven Replay Across Model–Domain Settings

C.3.1 RHO Loss Dynamics

Figure 8 extends the RHO-score dynamics shown in the main paper to the remaining model–domain settings. Across all settings, we observe the same qualitative feedback loop underlying RoD. At the beginning of training, adaptation examples have high RHO scores because substantial learning

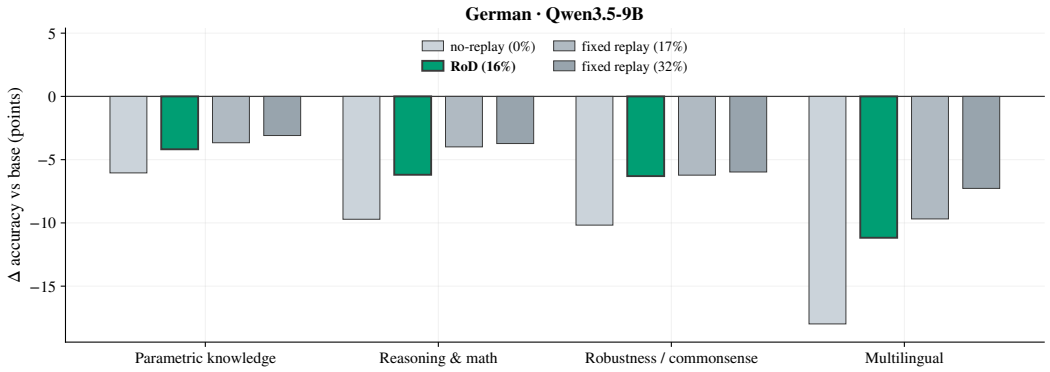


Figure 7: **Task-based forgetting with proxy replay.** Change in capability accuracy relative to the pretrained Qwen3.5-9B base model after German adaptation; values closer to zero indicate stronger retention. RoD substantially reduces forgetting relative to no-replay CPT across all capability groups. Its remaining gap to fixed replay is concentrated primarily in reasoning and mathematics and, to a lesser extent, multilingual evaluations, many of which contain translated reasoning and mathematical tasks.

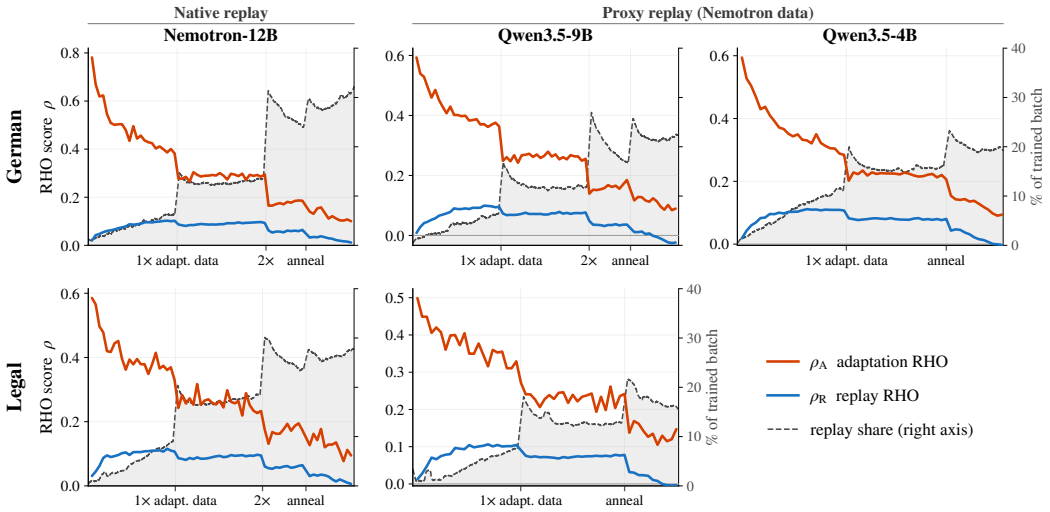


Figure 8: **RHO-score dynamics across model–domain settings.** Mean adaptation RHO score ρ_A and replay RHO score ρ_R over training, together with the resulting replay share. Across settings, adaptation scores decrease as the model learns the adaptation distribution, while replay scores initially increase as forgetting accumulates. The resulting competition between both signals dynamically adjusts the amount of replay throughout training. Vertical changes in replay share coincide with transitions between data passes and the learning-rate annealing phase.

potential remains, whereas replay scores start near zero, as the current model is still close to the pretrained model. As training progresses, adaptation scores decrease as the model learns the adaptation distribution, while replay scores increase as the model begins to forget. Replay therefore becomes increasingly competitive in the joint selection, causing the replay share to grow over training.

The abrupt changes in replay share coincide with transitions between passes over the adaptation data and the onset of learning-rate annealing. At these points, previously seen adaptation examples re-enter the candidate stream, temporarily changing their RHO-score distribution and thereby the balance between adaptation and replay. Despite these common dynamics, the resulting replay curricula differ across models and domains. For example, the replay share rises more sharply for Nemotron than for Qwen across several phases, and its trajectory differs between the German and

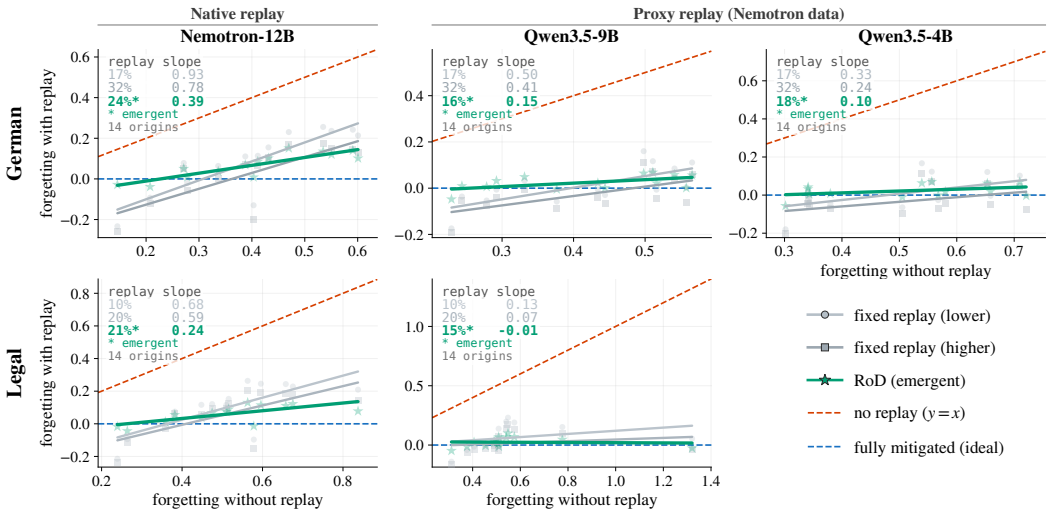


Figure 9: **Source-level forgetting profiles across model–domain settings.** Each point represents one replay-data source. The x-axis measures source-level forgetting under no-replay CPT, capturing how vulnerable a source is to forgetting, while the y-axis measures the forgetting that remains after applying replay. Lines show linear fits for fixed-replay CPT and RoD; the reported slopes summarize how strongly remaining forgetting depends on the source's original vulnerability. A slope near one indicates approximately uniform reduction of the no-replay forgetting profile, whereas a flatter slope indicates stronger relative protection of sources that would otherwise forget most. Across settings, RoD consistently produces the flattest forgetting profile.

Legal adaptations. Thus, RoD consistently exhibits the intended feedback mechanism while adapting the resulting amount of replay to the forgetting dynamics of each model–domain setting.

C.3.2 Forgetting Profiles

To examine whether the source-level forgetting patterns observed in the main paper generalize across settings, Figure 9 reports the corresponding profiles for the remaining model–domain combinations. For each replay source, we use forgetting under no-replay CPT as a measure of its vulnerability to continual adaptation (x-axis) and compare it against the forgetting remaining after replay (y-axis). The slope of the resulting forgetting profile captures how strongly this initial vulnerability persists after replay. A slope close to one indicates that replay approximately shifts the no-replay profile downward while preserving the relative differences between sources; a flatter slope indicates that sources prone to forgetting receive disproportionately stronger protection.

We observe the same qualitative pattern across models and adaptation domains. Fixed-replay CPT reduces overall forgetting, but largely preserves the underlying forgetting profile: sources that forget most without replay generally remain those with the largest residual forgetting. Increasing the fixed replay share pushes this profile further downward and moderately reduces its slope. RoD changes the profile more substantially. Across all settings, it yields the flattest fitted slope, preferentially reducing forgetting for vulnerable sources while spending less replay capacity on sources that already remain stable. For example, for German adaptation the slope decreases from $0.93/0.78$ under fixed replay to 0.39 with RoD for Nemotron-12B, from $0.50/0.41$ to 0.15 for Qwen3.5-9B, and from $0.33/0.24$ to 0.10 for Qwen3.5-4B. The same behavior appears for Legal adaptation, where RoD reduces the slope to 0.06 for Nemotron-12B.

The Legal–Qwen3.5-9B setting contains one influential multilingual knowledge category that substantially flattens all fitted profiles. We retain this knowledge category in fig. 9, but verify that the result is not driven by this outlier. Excluding it increases the slopes of the two fixed-replay profiles from 0.13 and 0.07 to 0.74 and 0.65, respectively, while the RoD slope increases from -0.01 to only 0.23. Thus, the same pattern becomes clearer after excluding the outlier: fixed replay primarily reduces the magnitude of forgetting while retaining much of the source-level vulnerability profile,

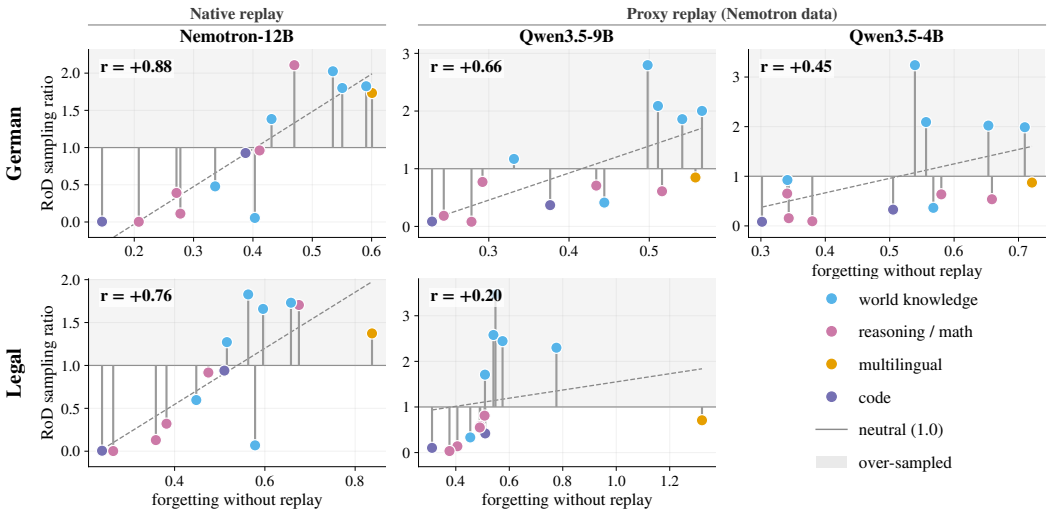


Figure 10: **Replay allocation follows source-level forgetting across model–domain settings.** Each point represents one replay-data source. The x-axis measures source vulnerability as forgetting under no-replay CPT, while the y-axis shows its sampling ratio under RoD relative to its prevalence in the replay candidate distribution; values above 1 indicate oversampling and values below 1 undersampling. Colors denote broad pretraining-data categories. Dashed lines show linear fits and r denotes the Pearson correlation between vulnerability and RoD sampling ratio. Across settings, RoD preferentially allocates replay to sources that are more susceptible to forgetting.

whereas RoD more strongly redistributes protection toward the sources that are most susceptible to forgetting.

Overall, these results show that the demand-driven behavior observed for German adaptation with Nemotron-12B generalizes across model families and domains. RoD does not merely determine how much replay to use; its joint selection also changes which parts of the pretraining distribution are protected, concentrating replay where forgetting is most pronounced.

C.3.3 Replay Allocation

The flatter forgetting profiles in section C.3.2 suggest that RoD allocates replay preferentially toward sources that are most vulnerable to forgetting. Figure 10 directly tests this mechanism across the remaining model–domain settings. For each replay source, we compare its vulnerability (measured as forgetting under no-replay CPT) against its sampling ratio under RoD. A ratio of 1 corresponds to sampling proportional to the source's prevalence in the replay candidate distribution, while values above or below 1 indicate increased or decreased sampling frequency, respectively.

Across settings, replay allocation is positively associated with source vulnerability. The relationship is strongest for German adaptation with Nemotron-12B ($r = 0.88$), but remains positive for Qwen3.5-9B ($r = 0.66$), Qwen3.5-4B ($r = 0.45$), and Legal adaptation with Nemotron-12B ($r = 0.76$). Sources that remain comparatively stable are often sampled below their prevalence in the replay distribution, whereas highly vulnerable sources can be replayed at more than twice their baseline rate. Thus, the source-level protection observed in section C.3.2 arises from an adaptive redistribution of replay rather than from uniformly increasing replay across the pretraining distribution.

As in the forgetting-profile analysis, the Legal–Qwen3.5-9B setting contains an influential multilingual source. Including this source yields a weaker correlation of $r = 0.20$; excluding it increases the correlation to $r = 0.69$ across the remaining 13 sources.

Together, these results show that the demand-driven allocation observed in the main-paper setting generalizes across models and domains: RoD uses the emerging forgetting signal not only to deter-

mine *how much* to replay, but also *what* to replay, concentrating its replay budget on the parts of the pretraining distribution that currently require it most.

D Limitations and Conceptual Considerations

D.1 Computational Requirements

RoD introduces additional computation relative to standard CPT through both online candidate scoring and offline reference construction. During training, each optimization step first scores a candidate pool of mk examples using a forward-only pass, before applying the standard forward–backward update to the selected batch of k examples. Candidate scoring requires no gradients or optimizer updates, but its cost grows linearly with the candidate multiplier m. In our experiments, we use $m = 2$, such that $2k$ candidates are evaluated for every k examples used for training. Our ablations further show little benefit from increasing the candidate pool beyond $m = 2$, limiting the additional online computation required in our experiments.

RoD additionally requires reference losses for both candidate streams. For replay, the reference is the pretrained model itself; for adaptation, our implementation constructs a specialized reference by training on the adaptation corpus until convergence. In our experiments, this specialist also serves as the no-replay CPT baseline and therefore does not require a separate training run within the experimental suite. In a standalone RoD application, however, constructing such a reference incurs an additional offline cost unless a suitable specialist is already available. Reference losses are fixed and can be precomputed and cached, such that neither reference model needs to be evaluated during RoD training. Moreover, the adaptation reference need not match the scale of the model being adapted: our cross-scale ablation shows that a 4B reference recovers a similar adaptation–forgetting trade-off for a 9B target model.

Our implementation is designed primarily to study whether dynamic, loss-based replay allocation is effective and to characterize the resulting data curriculum, rather than to optimize computational efficiency. Importantly, our cross-scale experiments provide a practical route to decoupling RoD's computational requirements from the target model's scale. A curriculum constructed by a smaller model can be transferred to a larger target model, avoiding candidate scoring and model-dependent reference-loss computation at the target model's scale. Together with the smaller adaptation reference above, these results suggest that RoD's model-dependent computation need not scale directly with the model being adapted. Further reducing the cost of curriculum construction remains an important direction for future work.

D.2 Relative Weighting of Adaptation and Replay

RoD directly compares adaptation and replay scores in the same token-normalized loss space, placing both signals on equal footing during joint selection. This provides a parameter-free default in which the adaptation–retention trade-off emerges from the model's current learning and forgetting state rather than from a predefined preference. For applications that require a specific operating point, the framework could be extended with a relative weighting between the two signals to explicitly favor adaptation or retention. We leave this extension to future work.

D.3 Requirements on the Replay Distribution

RoD can only protect knowledge that is represented in the available replay distribution. While the method determines which replay examples to prioritize and how much replay to allocate during training, it assumes access to a replay pool that sufficiently covers the parts of the pretraining distribution that should be retained. If relevant knowledge or capabilities are absent from this pool, their degradation cannot be detected through the corresponding replay examples, and RoD cannot selectively allocate replay toward them. Thus, RoD removes the need to prescribe the composition and amount of replay within an available replay distribution, but does not remove the requirement for representative replay data.

Our Qwen3.5 experiments illustrate that this requirement does not imply access to the exact original pretraining corpus. Because Qwen's pretraining data are unavailable, we use the Nemotron corpus

as proxy replay and still observe substantial reductions in forgetting. At the same time, the larger discrepancy between validation loss and task-based forgetting in this setting indicates that mismatched replay data can yield a less faithful retention signal. We therefore expect RoD to be most effective when the replay distribution provides broad coverage of the knowledge and capabilities that should be retained, while representative proxy data remain a viable alternative when the original pretraining distribution is unavailable.

D.4 Interpretation and Scope of the Replay Signal

RoD operationalizes forgetting through increases in language-modeling loss relative to the pretrained model, $\rho_R(x;\theta_t) = \ell_{\theta_t}(x) - \ell_{\theta_0}(x)$. This provides a task-agnostic signal that can be evaluated directly on replay examples during CPT, without requiring downstream labels or capability-specific evaluations. However, sequence-level loss degradation is not equivalent to downstream capability degradation. RoD can therefore react only to forgetting that manifests as increased loss on the available replay data, and the strength of this correspondence depends on how well those data represent the capabilities of interest.

This distinction is most visible in our proxy-replay experiments. For Nemotron, replay examples originate from the model's own pretraining distribution, providing a direct reference for degradation from the pretrained state. For Qwen3.5, loss changes are instead measured using the Nemotron proxy distribution, and we observe a weaker correspondence between validation loss and task-based forgetting, particularly for reasoning-intensive capabilities. Nevertheless, RoD substantially reduces task-based forgetting relative to no-replay CPT in these settings, suggesting that the loss-based signal remains useful even when this correspondence is imperfect. More generally, our results support base-relative loss as a practical online signal for demand-driven replay, while its interpretation as capability forgetting should remain tied to the coverage and alignment of the replay distribution.