# DELETE ONCE MEANS ONCE: REQUEST IDEMPOTENCE IN APPROXIMATE MACHINE UNLEARNING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Machine unlearning is typically evaluated as a one-shot model transformation, but deployed deletion services must also satisfy request-level correctness. We study *idempotence*: once a deletion request succeeds, replaying the identical request should be a no-op because $(D \setminus A) \setminus A = D \setminus A$. This guarantee does not follow automatically from approximate unlearning implemented as stateless finite-step optimization. On CIFAR-100/ResNet-18, a second identical NegGrad+ request produces $4.60\times$ the predictive variation of independent single-request runs (95% hierarchical-bootstrap CI $[4.42, 4.79]$, permutation $p = 0.0038$), with similar amplification across three independently trained models. A compute-matched retain-only continuation does not reproduce the effect, while applying NegGrad+ to an exact retraining endpoint moves the model by $0.09497$ JS divergence and reduces full-test accuracy to $67.67\%$. Gradient measurements show that the forgetting surrogate remains active at the exact target. Across five vision settings, duplicate amplification ranges from $1.49\times$ to $6.40\times$; a MovieLens-100K/BPR-MF test shows a weaker $1.69\times$ effect without ranking-utility loss. In a representative long-horizon seed-0 trajectory, NegGrad+ transitions from gradual drift to near-maximal predictive disagreement and chance-level accuracy around the seventh identical request. We argue that request idempotence should be an explicit machine-unlearning service contract and demonstrate a lightweight stateful wrapper that enforces it by construction.

## 1 INTRODUCTION

Machine unlearning asks a trained model to remove the influence of selected training data without full retraining (Cao & Yang, 2015; Ginart et al., 2019; Bourtoule et al., 2021). Modern approximate methods make this practical for deep networks through fine-tuning, gradient reversal, distillation, parameter damping, or salient-weight updates (Kurmanji et al., 2023; Fan et al., 2024; Foster et al., 2024). Evaluation has therefore focused on a one-shot question: after processing deletion set $A$, how close is the model to a reference such as retraining on $D \setminus A$?

We ask a different question: *what happens when the same successful deletion request is delivered twice?* Duplicate requests are routine at service boundaries: timeouts, job restarts, queue redelivery, reconciliation, and repeated user actions can replay completed operations. At the data level, deletion is idempotent:

$$(D \setminus A) \setminus A = D \setminus A. \tag{1}$$

After the first request, neither the desired training data nor the deletion target changes, so a reliable deletion interface has no new semantic work to do.

Approximate unlearning algorithms, however, are usually exposed as model-update procedures rather than stateful deletion services. If $\mathcal{U}_A$ means "run $m$ additional optimization steps using a forgetting surrogate for $A$," replaying the request invokes another $m$ steps, with no general reason for $\mathcal{U}_A(\mathcal{U}_A(\theta))$ to equal $\mathcal{U}_A(\theta)$. That an optimizer changes a model when run twice is trivial; the key question is whether this mismatch survives controls for ordinary optimization stochasticity and generic extra optimization.

We separate these effects empirically and mechanistically. In our core CIFAR-100/ResNet-18 setting, a second identical NegGrad+ request produces $4.60\times$ the predictive variation between independent



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.

single-request runs. The effect is statistically separated from stochasticity and appears across three independently trained base models. A compute-matched retain-only continuation leaves the distance to exact retraining almost unchanged, whereas repeating the forgetting objective increases it. More strongly, an exact RETRAIN-A model is not a NegGrad+ fixed point: applying the same operator to a model trained from scratch on $D \setminus A$ causes substantial drift. Gradient measurements explain why: the forgetting component remains large at the exact target, so the surrogate still exerts update pressure after the deletion semantics are satisfied.

Our experiments also clarify scope. Duplicate sensitivity is strong but not universal. Across five image-classification settings, second-request amplification ranges from $1.49$ to $6.40$; MovieLens-100K/BPR-MF shows weaker excess drift ($1.69\times$) without ranking-utility degradation. In a representative seed-0 trajectory, repeated NegGrad+ replay in the core setting eventually causes a sharp high-confidence collapse: around the seventh identical request, full-test accuracy approaches chance while predictive JS divergence approaches its theoretical maximum.

These findings separate *unlearning algorithms* from *unlearning service semantics*. An approximate optimizer need not be idempotent, but a stateful service can be. Our minimal reference wrapper, DELETEGUARD, records a durable idempotency key and returns the committed result on duplicate delivery. In a single-database implementation, 10,000 duplicates cause zero additional unlearning calls with microsecond-scale lookup overhead. DELETEGUARD is not a new unlearning algorithm; it demonstrates that request idempotence is best treated as an explicit interface contract.

Our contributions are:

- We formulate **request-level idempotence** as a machine-unlearning service property and introduce a stochasticity-aware measure of duplicate amplification.
- We show that approximate unlearning can violate this property: a second identical NegGrad+ request reaches $4.60\times$ the single-request stochastic baseline in the core setting, with similar amplification across training seeds.
- We rule out a simple extra-compute explanation, show that exact retraining is not a NegGrad+ fixed point, and connect the failure to a non-vanishing forgetting surrogate at the exact target.
- We characterize boundary conditions across methods, datasets, architectures, request sizes, and a recommender setting, and show that long-horizon duplicate replay can trigger catastrophic high-confidence collapse.
- We provide a lightweight stateful design that enforces duplicate-request idempotence at the service boundary.

## 2 RELATED WORK

**Machine unlearning.** Early work introduced machine unlearning as the problem of removing data and its downstream influence from learned systems (Cao & Yang, 2015), while later work formalized efficient data deletion and exact or partitioned retraining strategies (Ginart et al., 2019; Bourtoule et al., 2021). For deep networks, approximate methods trade exact equivalence for efficiency. SCRUB uses a teacher–student objective to selectively forget while retaining utility, and its evaluation also introduced the widely used NegGrad+ baseline (Kurmanji et al., 2023). SalUn focuses updates on gradient-salient weights (Fan et al., 2024); selective synaptic dampening uses relative parameter importance to reduce reliance on forget data without retain-set retraining (Foster et al., 2024). Our work does not propose another unlearning objective. Instead, we study what happens when such objectives are invoked repeatedly under an unchanged deletion target.

**Evaluation and auditing.** Unlearning evaluation is difficult because behavioral similarity to retraining does not certify training history. Thudi et al. (2021) argue for auditable algorithmic definitions. One-shot evaluations can also miss failures: Pawelczyk et al. (2025) show that practical methods may fail to remove data-poisoning effects, while benchmarks reveal sensitivity to methods, hyperparameters, architectures, and initializations (Cadet et al., 2025). Lanyon et al. (2025) show that single-training-seed evaluation can mislead. We therefore use multiple unlearning runs and independently trained base models.

**Continual and repeated unlearning.** Continual unlearning studies sequences of deletion tasks and the ability to keep forgetting as the target state evolves. Recent work identifies plasticity collapse under repeated unlearning operations, including reduced ability to forget future tasks and re-emergence of previously forgotten information (Shi et al., 2026). Duplicate requests are a sequence-level special case, but they isolate a distinct semantic regime: after the first $A$, the second $A$ adds no new deleted records and the desired target is exactly unchanged. We therefore do not claim that duplicate requests are unrelated to continual unlearning. Rather, idempotence is the zero-target-change edge case that a deletion service should handle safely. Section 5.3 empirically contrasts $A \to A$ with a matched $A \to B$ sequence.

# 3 Request-Level Idempotence

## 3.1 Deletion semantics

Let $D$ be a training dataset, $A \subseteq D$ a deletion request, and $\theta_0$ the model before deletion. The exact retraining target for $A$ is a model produced by running the training procedure on

$$R_A = D \setminus A. \tag{2}$$

At the service level, maintain a processed-deletion state $H \subseteq D$. A new request $A$ updates this state as

$$H' = H \cup A. \tag{3}$$

If $A \subseteq H$, then $H' = H$ and the target dataset is unchanged. This gives the following basic contract.

**Request-idempotence contract.** After a successful request $A$, replaying the identical request should produce no new semantic state transition. In deterministic service terms,

$$\mathrm{Delete}(\mathrm{Delete}(S, A), A) = \mathrm{Delete}(S, A), \tag{4}$$

where $S$ includes both model state and deletion metadata.

Approximate unlearning procedures are usually represented without the metadata component. Let $\mathcal{U}_A(\theta)$ denote one invocation of an approximate unlearning algorithm for $A$. Exact parameter equality is an inappropriate requirement for stochastic algorithms, so we evaluate a behavioral analogue:

$$d(\mathcal{U}_A(\mathcal{U}_A(\theta_0)), \mathcal{U}_A(\theta_0)) \tag{5}$$

should be no larger than variation ordinarily induced by independent realizations of a single request.

## 3.2 Stochasticity-aware duplicate amplification

Let $\mathcal{U}_A^{(i)}(\theta_0)$ be an independently randomized single-request endpoint and let $\mathcal{U}_A^{(2,i)}(\theta_0)$ denote the endpoint after two identical requests in run $i$. For a predictive distance $d$, define the single-request stochastic baseline

$$V_{\mathrm{stoch}} = \mathbb{E}_{i \neq j}\left[d\Big(\mathcal{U}_A^{(i)}(\theta_0), \mathcal{U}_A^{(j)}(\theta_0)\Big)\right], \tag{6}$$

and duplicate-request drift

$$V_{\mathrm{dup}} = \mathbb{E}_{i,j}\left[d\Big(\mathcal{U}_A^{(2,i)}(\theta_0), \mathcal{U}_A^{(j)}(\theta_0)\Big)\right]. \tag{7}$$

We report the *Duplicate Amplification Ratio*

$$\mathrm{DAR}_2 = \frac{V_{\mathrm{dup}}}{V_{\mathrm{stoch}}}. \tag{8}$$

$\mathrm{DAR}_2 = 1$ means that the duplicate effect is at the scale of ordinary single-request stochasticity. We do not posit an arbitrary universal threshold above one; the ratio is a normalized severity measure, interpreted together with absolute drift and utility.

Our main distance is mean per-example symmetric Jensen–Shannon divergence between the two softmax predictive distributions:

$$d(P, Q) = \frac{1}{|\mathcal{T}|}\sum_{x \in \mathcal{T}} \frac{1}{2} D_{\mathrm{KL}}(P_x \| M_x) + \frac{1}{2} D_{\mathrm{KL}}(Q_x \| M_x), \quad M_x = \frac{P_x + Q_x}{2}, \tag{9}$$

using natural logarithms. Thus $d \leq \ln 2$.

### 3.3 WHY FINITE-STEP UNLEARNING NEED NOT BE IDEMPOTENT

Many practical methods are finite-step iterative procedures. Suppose one invocation runs $m$ updates of a map $T_A$:

$$\mathcal{U}_A(\theta) = T_A^m(\theta). \tag{10}$$

Then two identical requests produce $T_A^{2m}(\theta)$. Request idempotence therefore requires

$$T_A^{2m}(\theta) = T_A^m(\theta), \tag{11}$$

which is guaranteed only when the first endpoint is already a fixed point of the $m$-step operator (or when later updates cancel exactly). A fixed epoch budget does not generally have this property.

The mismatch is especially transparent for gradient-based forgetting objectives. NegGrad+ optimizes a surrogate of the form

$$J_A(\theta) = \beta L_{R_A}(\theta) - (1-\beta)L_A(\theta). \tag{12}$$

Let $\theta_R^\star$ be an exact retraining endpoint trained on $R_A$. Even if $\nabla L_{R_A}(\theta_R^\star)$ is small, the unlearning update at this already-deleted target is

$$\nabla J_A(\theta_R^\star) = \beta \nabla L_{R_A}(\theta_R^\star) - (1-\beta)\nabla L_A(\theta_R^\star), \tag{13}$$

which need not vanish. Thus exact deletion semantics and the fixed points of the approximate forgetting surrogate can differ.

## 4 EXPERIMENTAL PROTOCOL

### 4.1 CORE IMAGE-CLASSIFICATION SETTING

Our core experiment uses CIFAR-100 (Krizhevsky, 2009) with ResNet-18 (He et al., 2016). The duplicate request $A$ contains 5,000 of the 50,000 training examples, so the request fraction is 10%. We emphasize this definition because some artifact directory names encode the 20% union $A \cup B$ used in a separate two-request experiment; throughout the paper, *request fraction* means $|A|/|D|$.

We compare every approximate endpoint against a fixed RETRAIN-A oracle trained from scratch on $D \setminus A$. For the main stochasticity analysis, we run five independent unlearning realizations from the same trained base model. We additionally repeat the second-request test on two independently trained base models, for three training seeds in total.

### 4.2 METHODS

We study three representative update styles. **Fine-tune** continues optimization on retained data only. **NegGrad+** optimizes Equation 12 with $\beta = 0.95$. **SalUn** uses gradient-based parameter saliency to restrict the unlearning update (Fan et al., 2024). Core unlearning invocations use 10 epochs; implementation and hyperparameters are provided in the supplementary code and summarized in Appendix A. Retraining is used as an oracle, not as a proposed deployment method.

### 4.3 METRICS AND STATISTICAL TESTS

Our primary behavioral distance is full-test predictive JS divergence. Utility is full held-out test accuracy; train-set retain accuracy is treated only as a diagnostic and is not used as the main utility metric. We report single-request stochasticity, duplicate drift, $\mathrm{DAR}_k$, and distance to RETRAIN-A. For the core $\mathrm{DAR}_2$ result we use a 5,000-sample hierarchical bootstrap for the confidence interval and a 5,000-draw run-label permutation test. We also report cross-training-seed replication in accordance with recent concerns about single-seed unlearning evaluation (Lanyon et al., 2025).

### 4.4 BREADTH, CROSS-DOMAIN, AND LONG-HORIZON TESTS

For image breadth, we test NegGrad+ on five settings: CIFAR-100/ResNet-18 with 5%, 10%, and 20% requests; CIFAR-100/VGG-16 (Simonyan & Zisserman, 2015) with a 10% request; and CIFAR-10/ResNet-18 with a 10% request. Each breadth cell uses three unlearning realizations.

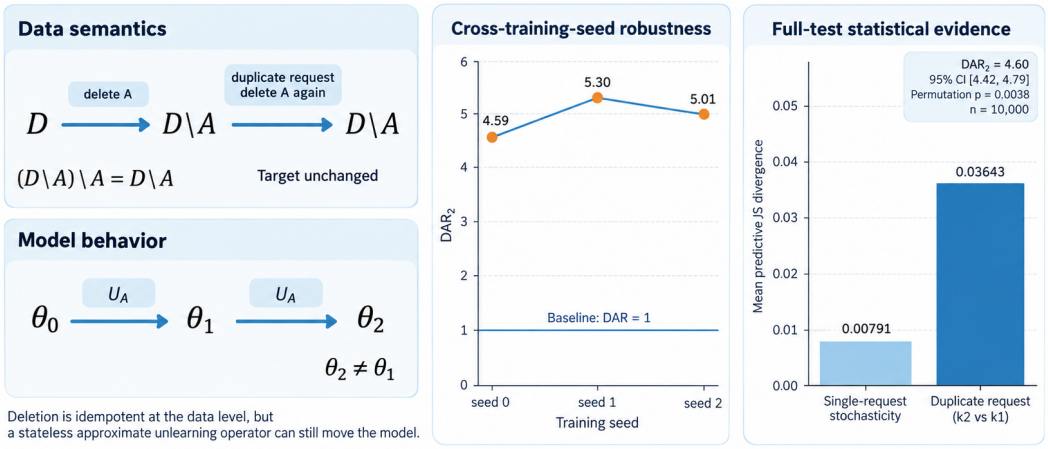


Figure 1: Identical deletion requests can change the model even when the deletion target is unchanged. Left: the data-level idempotence contract. Middle: cross-training-seed $\mathrm{DAR}_2$. Right: the full-test stochasticity comparison and statistical result.

For a cross-domain boundary test, we use MovieLens-100K (Harper & Konstan, 2015) with a 64-dimensional Bayesian Personalized Ranking matrix-factorization model (Rendle et al., 2009). Ratings at least four are treated as positive implicit feedback, the latest positive per user is held out, and 5% of training interactions are forgotten. Predictive distance is computed over a fixed candidate set of one held-out positive and 99 negatives per user; utility is Recall@20 and NDCG@20.

Finally, we replay the same NegGrad+ request up to $k = 10$ in the core seed-0 setting and track distance to the first successful deletion, distance to RETRAIN-A, full-test accuracy, mean logit norm, and predictive entropy.

## 5 RESULTS

### 5.1 A SECOND IDENTICAL REQUEST EXCEEDS ORDINARY STOCHASTICITY

Figure 1 summarizes the core phenomenon. With five independent unlearning realizations, the mean single-request stochastic variation is

$$V_{\mathrm{stoch}} = 0.007911, \tag{14}$$

while the second identical request produces

$$V_{\mathrm{dup}} = 0.036426, \qquad \mathrm{DAR}_2 = 4.604. \tag{15}$$

The 95% hierarchical-bootstrap interval for $\mathrm{DAR}_2$ is $[4.421, 4.792]$, and the run-label permutation test gives $p = 0.003799$. The absolute excess drift, $V_{\mathrm{dup}} - V_{\mathrm{stoch}} = 0.028515$, has 95% interval $[0.02749, 0.02957]$.

The result is not specific to one trained model. Across three independent training seeds, the second-request amplification is approximately $4.59\times$, $5.30\times$, and $5.01\times$. We therefore treat the core effect as a reproducible duplicate-request sensitivity rather than an artifact of one initialization.

### 5.2 THE EFFECT IS NOT GENERIC EXTRA OPTIMIZATION

A reviewer could reasonably argue that a second request merely grants the model more optimization steps. Figure 2 directly tests this explanation. After the first NegGrad+ request, mean full-test JS distance to RETRAIN-A is $0.08572$. Spending the same additional 10-epoch budget on retained data only changes the distance to $0.08564$, whereas running the duplicate NegGrad+ request increases it to $0.10348$. Mean full-test accuracy is $76.69\%$ after the first request, $76.76\%$ after retain-only continuation, and $74.29\%$ after the duplicate request. Thus extra compute by itself does not reproduce the failure.

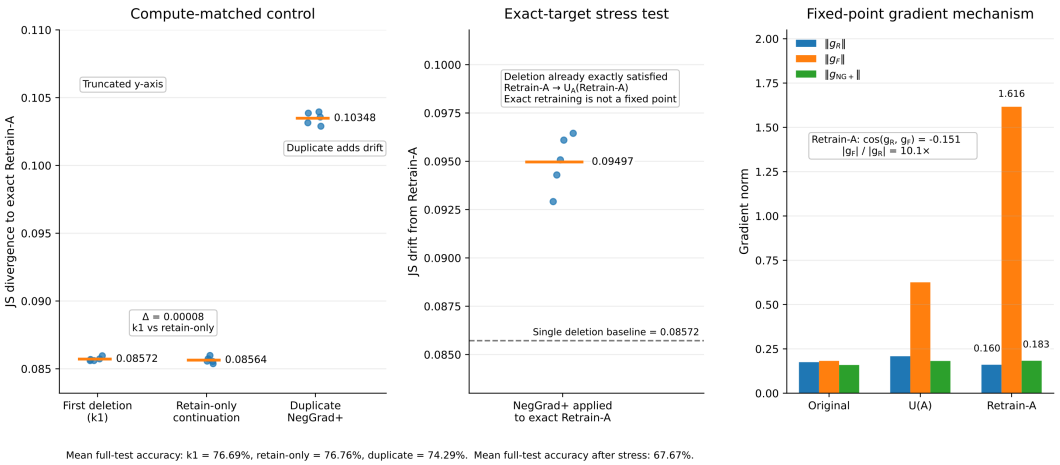


Figure 2: Duplicate-request drift is driven by the forgetting objective, not merely by additional optimization. The panels compare a compute-matched retain-only continuation with duplicate NegGrad+, apply NegGrad+ to an exact retraining endpoint, and report gradient norms at Original, $U(A)$, and RETRAIN-A.

Table 1: Duplicate versus novel second requests. Target change is the predictive change between the corresponding exact retraining targets; model movement is the second-step endpoint change.

| Second request | New deleted | Target change JS | Model movement JS | Test acc. after |
|---|---|---|---|---|
| $A \to A$ (duplicate) | 0 | 0.00000 | **0.03637** | 74.39% |
| $A \to B$ (novel) | 5,000 | 0.09144 | 0.01197 | 76.42% |

We next start from an exact RETRAIN-A checkpoint, for which the requested data $A$ were never used in training. Applying NegGrad+ for the same $A$ nevertheless moves the model by mean JS $0.09497$ from the retrained endpoint and reduces full-test accuracy to $67.67\%$. In this concrete sense, an exact retraining endpoint is not a fixed point of the approximate forgetting operator.

The gradient measurements are consistent with Equation 12. At RETRAIN-A, the retain-gradient norm is $0.160$, while the forget-gradient norm is $1.616$, about $10.1\times$ larger. Their cosine similarity is $-0.151$, and the resulting NegGrad+ objective gradient norm remains $0.183$. We do *not* claim that the finite-training RETRAIN-A checkpoint has zero retain gradient. The relevant observation is that the forgetting surrogate remains strongly active even after the deletion semantics are already exactly satisfied.

### 5.3 DUPLICATE REQUESTS ISOLATE A ZERO-TARGET-CHANGE REGIME

Duplicate delivery is a repeated-unlearning sequence, so we explicitly compare it with a genuinely new second request. We construct a matched $A \to A$ versus $A \to B$ experiment, where $B$ contains 5,000 new points.

Table 1 shows the semantic distinction directly. The duplicate request changes the target by exactly zero but moves the model by $0.03637$ JS. The novel request changes the exact target by $0.09144$ JS but moves the model only $0.01197$ in the second step. Thus the duplicate case produces roughly $3.0\times$ more model movement despite zero target change. This does not make duplicate unlearning unrelated to continual unlearning; rather, it identifies idempotence as a particularly clean edge case in which any systematic second-step movement is semantically unnecessary.

### 5.4 THE FAILURE IS METHOD-DEPENDENT

Table 2 shows why stochastic normalization matters. Fine-tuning has little second-request amplification, although drift accumulates gradually by $k = 4$. NegGrad+ shows strong amplification

Table 2: Request-idempotence behavior is strongly method-dependent on CIFAR-100/ResNet-18 with a 10% request and five optimization replicates. Test accuracy is evaluated on the full held-out test set.

| Method | Single-run noise JS | $DAR_2$ | $DAR_3$ | $DAR_4$ | Test acc. $k1 \to k4$ | JS to RETRAIN-A $k1 \to k4$ |
|---|---|---|---|---|---|---|
| Fine-tune | 0.00696 | 1.15 | 1.51 | 2.10 | 76.74% → 76.86% | 0.0857 → 0.0873 |
| NegGrad+ | 0.00798 | **4.59** | **13.64** | **18.37** | 76.69% → 65.29% | 0.0858 → 0.1711 |
| SalUn | 0.19872 | 1.06 | 1.20 | 1.31 | 64.35% → 56.26% | 0.1787 → 0.2403 |

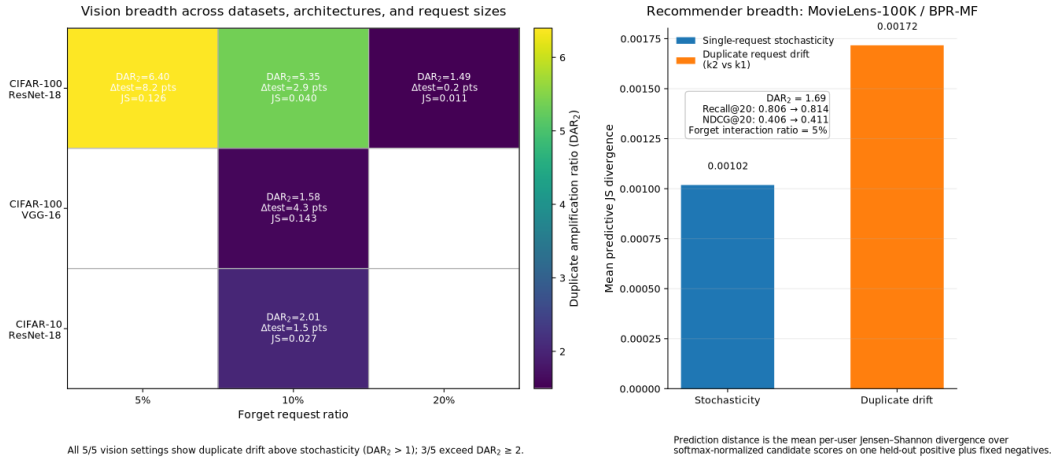


Figure 3: Duplicate-request drift generalizes beyond the main CIFAR-100/ResNet-18 setting, but severity is regime-dependent. Left: a vision breadth matrix over datasets, architectures, and request sizes. Right: a MovieLens-100K/BPR-MF cross-domain test.

immediately and progressively moves farther from exact retraining. SalUn exhibits large absolute variation and utility degradation, but its *single-request* stochastic baseline is already extremely high (0.19872 JS), so its DAR remains close to one. Our conclusion is therefore not that every approximate method has the same idempotence failure. The severity depends on the operator and its intrinsic variance.

### 5.5 BREADTH AND BOUNDARY CONDITIONS

Figure 3 expands the evaluation beyond the core cell. For CIFAR-100/ResNet-18, the independent breadth sweep gives $DAR_2 = 6.40$, 5.35, and 1.49 for 5%, 10%, and 20% requests respectively. CIFAR-100/VGG-16 at 10% gives 1.58, and CIFAR-10/ResNet-18 gives 2.01. All five vision settings have duplicate drift larger than their single-request stochastic baseline, but the magnitude is clearly non-monotonic in request size and sensitive to architecture.

The recommender result is a useful boundary case rather than a universal-failure claim. On MovieLens-100K/BPR-MF, single-request stochasticity is 0.001018 JS and duplicate drift is 0.001717, giving $DAR_2 = 1.69$. However, Recall@20 changes from 0.806 to 0.814 and NDCG@20 from 0.406 to 0.411, so the modest excess drift does not harm ranking utility in this setting. The combined result is that request sensitivity transfers beyond one image model, while its practical severity remains operator- and domain-dependent.

### 5.6 LONG-HORIZON REPLAY CAUSES A SHARP COLLAPSE

The second request is enough to falsify request idempotence, but long-horizon replay reveals the practical consequence of repeatedly violating the contract. Figure 4 shows a representative seed-0 trajectory. From $k = 1$ to $k = 6$, JS distance from the first deletion increases from 0 to 0.181, distance to RETRAIN-A increases from 0.086 to 0.195, and full-test accuracy decreases from 76.61% to 62.73%.

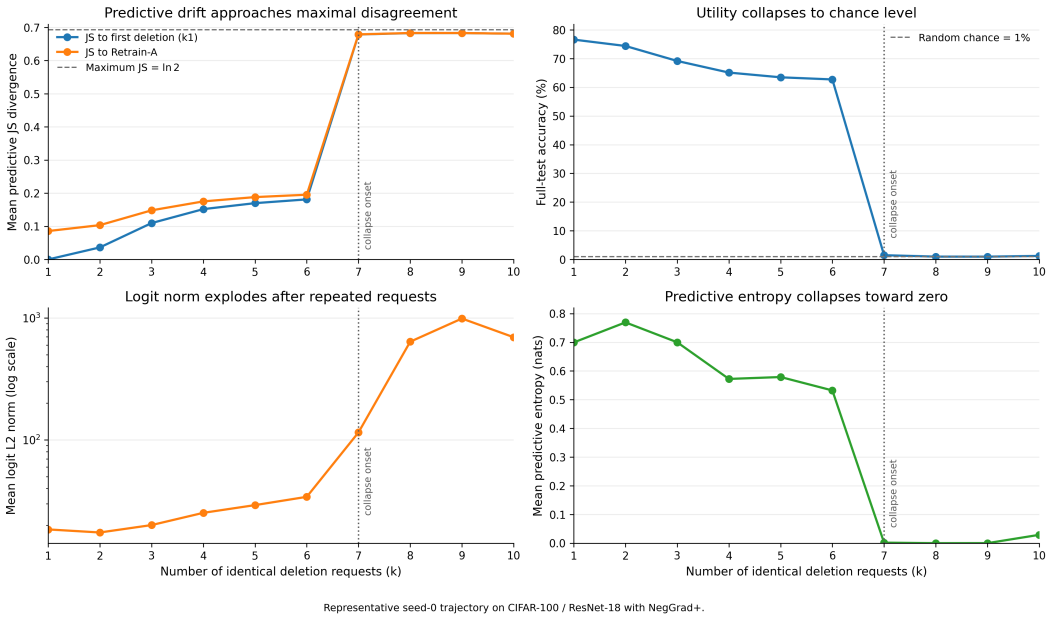


Figure 4: Repeated identical deletion requests transition from gradual drift to catastrophic high-confidence collapse. The panels plot JS distance, full-test accuracy, logit norm, and predictive entropy for a representative seed-0 NegGrad+ trajectory through $k = 10$.

At $k = 7$, the dynamics change abruptly: test accuracy falls to $1.48\%$ on the 100-class task, while JS distance jumps to $0.679$, close to the theoretical maximum $\ln 2 \approx 0.693$. This is not a transition to uniform uncertainty. Mean logit norm rises from $34.2$ at $k = 6$ to $115.2$ at $k = 7$, $638.5$ at $k = 8$, and $988.9$ at $k = 9$, while predictive entropy collapses from $0.532$ to $0.0018$ at $k = 7$ and effectively zero thereafter. Repeated forgetting therefore drives the model into an extremely high-confidence degenerate predictive regime.

We stress that the $k = 1 \ldots 10$ trajectory is a mechanism-oriented representative run rather than a multi-seed estimate of the collapse time. The robust claim is the second-request amplification; the long-horizon result illustrates one severe downstream trajectory when the same semantic error is repeatedly replayed.

## 6 Enforcing Idempotence at the Service Boundary

The preceding results suggest that request idempotence should not be delegated to the convergence behavior of an approximate optimizer. A service can enforce the property independently.

Our reference wrapper, DeleteGuard, associates each deletion request with a durable idempotency key and a payload hash. When a key is first processed, the underlying unlearning operation runs and its committed model version is recorded. If the same key and payload arrive again, the service returns the stored result without invoking unlearning. Reusing a key with a conflicting payload is rejected. In systems that track individual deleted identifiers, the same principle can be extended by maintaining the processed deletion set and applying the underlying unlearning algorithm only to newly requested identifiers.

In a SQLite-backed single-database reference implementation, 10,000 duplicate requests trigger zero additional underlying unlearning calls. Median duplicate lookup latency is $7.96\,\mu$s and the 95th percentile is $8.35\,\mu$s. The implementation rejects conflicting key reuse and, in a simulated pre-commit crash, atomically rolls back the request record and model-version update; retry then executes once.

This experiment is intentionally modest. We do not claim a production-grade distributed exactly-once protocol. If model storage and the idempotency registry reside in different failure domains, a production service would need an appropriate transactional, outbox, or equivalent commit protocol.

The role of DELETEGUARD is to demonstrate that the desired correctness contract is simple to state and can be enforced by service state even when the underlying unlearning algorithm is not idempotent.

## 7 DISCUSSION AND LIMITATIONS

**What the result does and does not show.** The central finding is a semantic mismatch: data deletion is idempotent, while a stateless approximate unlearning update need not be. DAR measures the behavioral severity of this mismatch relative to the method's own stochasticity. It is not a proof that a model satisfies a privacy definition, nor is $\mathrm{DAR} > 1$ a universal accept/reject threshold.

**Why the failure is not universal.** The method table and breadth experiments show substantial heterogeneity. Fine-tuning is comparatively stable under a second request; SalUn has high absolute variability that makes duplicate amplification small after normalization; large requests and the recommender setting also show weaker effects. This heterogeneity is scientifically useful: request idempotence is an interface requirement, but the consequence of violating it depends on the underlying algorithm.

**Scope of domains.** Our strongest controlled evidence is on image classification, with one cross-domain recommender experiment. We do not evaluate language-model or diffusion-model unlearning, where requests may target facts, concepts, or generated content rather than explicit training-example identifiers. Extending request-idempotence auditing to those domains is important future work.

**Exact retraining and stochastic targets.** Retraining itself is stochastic, and our fixed-point stress test uses a concrete RETRAIN-A realization. The claim is therefore that this exact retraining endpoint is not a fixed point of NegGrad+—not that every retraining realization would experience identical drift. A fuller study could compare against a distribution of independently retrained oracles.

**Long-horizon collapse.** The collapse around $k = 7$ is shown for one representative seed-0 trajectory. We use it as a mechanism and consequence study, not as an estimate of a universal "seven-request" threshold. The cross-seed claim in this paper concerns the second-request amplification.

**System semantics.** Exact duplicate keys are only one kind of semantic replay. Production systems may receive equivalent deletion payloads under different keys, overlapping requests, partial retries, or concurrent requests. Maintaining canonical deletion identities and durable processed-set state would broaden the guarantee. Our single-database DELETEGUARD prototype intentionally stops short of these distributed-systems questions.

## 8 CONCLUSION

Deleting the same data twice should not make a model forget twice. We formalize this intuition as request-level idempotence and show that it does not automatically emerge from stateless approximate unlearning optimizers. In a controlled CIFAR-100/ResNet-18 study, duplicate NegGrad+ requests induce predictive drift far beyond ordinary single-request stochasticity; matched-compute and exact-target controls connect the effect specifically to the forgetting objective, and gradient measurements expose a fixed-point mismatch between deletion semantics and the optimization surrogate. The effect generalizes with varying severity across several image settings and a recommender boundary case, and repeated replay can eventually cause catastrophic high-confidence collapse. These findings argue for a clean systems principle: approximate unlearning algorithms should be evaluated as components of a stateful deletion service, and request idempotence should be enforced as an explicit correctness contract rather than assumed as an emergent property of optimization.

### AI USE STATEMENT

In this work, we used generative AI tools for research brainstorming, drafting and editing prose, generating and debugging experimental code, and assisting with figure-layout design. All experiments were executed by the authors on their own compute environment; reported numerical results were taken from saved experimental outputs rather than generated by an AI system. The authors manually reviewed the experimental logic, reran debugging fixes, checked statistical outputs, and verified the related-work citations used in the manuscript. Generative AI was not used to fabricate experimental observations or to make autonomous decisions about the final scientific claims. 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 correctness and reliability properties of machine-unlearning systems, with the goal of reducing the risk that deletion requests are processed inconsistently. Experiments use standard public research benchmarks (CIFAR and MovieLens-100K) and do not involve new human-subject data collection or attempts to identify individual users. The results should not be interpreted as evidence that any evaluated approximate unlearning method satisfies a legal or privacy requirement; our metrics measure behavioral request consistency, not formal privacy compliance.

### REPRODUCIBILITY STATEMENT

The supplementary material will include the complete experiment code, request splits or their deterministic generation procedure, hyperparameters, evaluation scripts, bootstrap and permutation-test implementation, and the CSV files used to produce all paper figures and tables. The main paper specifies the request-level definitions, primary metrics, and core experimental settings; Appendix A summarizes implementation details and Appendix B reports the complete per-setting results. The final submission will provide an anonymous source-code archive sufficient to reproduce the reported experiments.

## REFERENCES

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy*, pp. 141–159, 2021. doi: 10.1109/SP40001.2021.00019.

Xavier F. Cadet, Anastasia Borovykh, Mohammad Malekzadeh, Sara Ahmadi-Abhari, and Hamed Haddadi. Deep unlearn: Benchmarking machine unlearning for image classification. In *10th IEEE European Symposium on Security and Privacy*, 2025.

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pp. 463–480, 2015. doi: 10.1109/SP.2015.35.

Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *International Conference on Learning Representations*, 2024.

Jack Foster, Stefan Schoepf, and Alexandra Brintrup. Fast machine unlearning without retraining through selective synaptic dampening. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 12043–12051, 2024. doi: 10.1609/aaai.v38i11.29092.

Antonio Ginart, Melody Y. Guan, Gregory Valiant, and James Zou. Making ai forget you: Data deletion in machine learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):1–19, 2015. doi: 10.1145/2827872.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Jamie Lanyon, Axel Finke, Petros Andreou, and Georgina Cosma. On the importance of multiple training seeds for evaluating machine unlearning, 2025.

Martin Pawelczyk, Jimmy Z. Di, Yiwei Lu, Gautam Kamath, Ayush Sekhari, and Seth Neel. Machine unlearning fails to remove data poisoning attacks. In *International Conference on Learning Representations*, 2025.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 452–461, 2009.

Yingdan Shi, Xiang Xu, Kaize Ding, Alfred O. Hero, and Ren Wang. On the plasticity collapse in continual machine unlearning, 2026.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Anvith Thudi, Hengrui Jia, Ilia Shumailov, and Nicolas Papernot. On the necessity of auditable algorithmic definitions for machine unlearning, 2021.

# A Implementation and hyperparameters

## A.1 Core image experiments

The core image setting uses CIFAR-100 with ResNet-18 and a 5,000-example (10%) request. Each unlearning invocation runs for 10 epochs. NegGrad+ uses learning rate $0.01$, weight decay $5 \times 10^{-4}$, gradient clipping at 5, and $\beta = 0.95$ in Equation 12. Fine-tune uses the same invocation length and optimizes only retained examples. SalUn uses the same base optimization scale together with saliency coefficient 1 and a 0.5 mask fraction. Mini-batch size is 256.

For stochastic controls we use five independent unlearning replicates. The seed-0 core setting is evaluated on all 10,000 CIFAR-100 test examples for the main JS statistics. Two additional independently trained ResNet-18 models are used for cross-training-seed replication of $k1$ and $k2$.

## A.2 Image breadth experiments

The breadth sweep uses three independent unlearning replicates per cell. Newly trained CIFAR base models use SGD with learning rate $0.1$, momentum $0.9$, weight decay $5 \times 10^{-4}$, and 150 epochs, with learning-rate reductions at approximately 60% and 80% of training. The duplicate test uses NegGrad+ for two identical requests with 10 unlearning epochs per request.

## A.3 MovieLens-100K experiment

We convert ratings $\geq 4$ into implicit positives. For each user with at least two positives, the latest positive is held out for ranking evaluation and earlier positives are used for training. The BPR matrix-factorization model has 64-dimensional user and item embeddings and is trained for 30 epochs with Adam at learning rate $10^{-3}$. The unlearning objective uses the same retain-minus-forget structure with $\beta = 0.95$, learning rate $5 \times 10^{-4}$, five epochs per request, and five stochastic replicates. Ranking is measured using one held-out positive and 99 fixed negative candidates per user.

# B ADDITIONAL QUANTITATIVE RESULTS

## B.1 IMAGE BREADTH TABLE

Table 3: Complete vision breadth results used in Figure 3. $\Delta$ test accuracy is the drop from $k1$ to $k2$ in percentage points.

| Dataset | Model | Request | Noise JS | $\mathrm{DAR}_2$ | $\Delta$ test acc. |
|---|---|---|---|---|---|
| CIFAR-100 | ResNet-18 | 5% | 0.01972 | 6.40 | $-8.25$ |
| CIFAR-100 | ResNet-18 | 10% | 0.00749 | 5.35 | $-2.89$ |
| CIFAR-100 | ResNet-18 | 20% | 0.00748 | 1.49 | $-0.21$ |
| CIFAR-100 | VGG-16 | 10% | 0.09086 | 1.58 | $-4.27$ |
| CIFAR-10 | ResNet-18 | 10% | 0.01331 | 2.01 | $-1.51$ |

The 10% CIFAR-100/ResNet-18 breadth cell is an independent three-replicate sweep and is therefore reported separately from the five-replicate core statistical result ($\mathrm{DAR}_2 = 4.60$).

## B.2 MOVIELENS-100K RESULT

Table 4: MovieLens-100K/BPR-MF boundary experiment.

| Noise JS | Duplicate JS | $\mathrm{DAR}_2$ | Recall@20 $k1$ | Recall@20 $k2$ | NDCG@20 $k1 \to k2$ |
|---|---|---|---|---|---|
| 0.001018 | 0.001717 | 1.69 | 0.806 | 0.814 | $0.406 \to 0.411$ |

## B.3 LONG-HORIZON NEGGRAD+ TRAJECTORY

Table 5: Representative seed-0 long-horizon trajectory.

| $k$ | JS to $k1$ | JS to RETRAIN-A | Test acc. | Forget-train acc. (diagnostic) |
|---|---|---|---|---|
| 1 | 0.000 | 0.086 | 76.61% | 98.90% |
| 2 | 0.036 | 0.104 | 74.39% | 84.06% |
| 3 | 0.110 | 0.148 | 69.17% | 57.84% |
| 4 | 0.152 | 0.175 | 65.12% | 41.06% |
| 5 | 0.170 | 0.188 | 63.47% | 33.26% |
| 6 | 0.181 | 0.195 | 62.73% | 30.32% |
| 7 | 0.679 | 0.679 | 1.48% | 0.06% |
| 8 | 0.683 | 0.683 | 0.99% | 0.10% |
| 9 | 0.683 | 0.683 | 0.98% | 0.24% |
| 10 | 0.681 | 0.681 | 1.23% | 0.48% |

Forget-training accuracy is shown only as a diagnostic. Lower accuracy on the requested examples is not interpreted as utility degradation.

# C STATEFUL IDEMPOTENCY REFERENCE

The reference implementation stores the model-version update and idempotency record in one SQLite transaction. This is sufficient to demonstrate local atomic semantics but is not a complete protocol for a distributed production model store.

# D STATISTICAL DEFINITIONS

For five single-request endpoints, the single-request stochastic baseline averages pairwise JS divergence over distinct $k1$ runs. Duplicate drift averages all $k2$-versus-$k1$ endpoint pairs. The main seed-0 statistical analysis evaluates predictions on the full CIFAR-100 test set. The reported confidence

Table 6: DELETEGUARD reference service semantics.

| Quantity | Result |
|---|---|
| Duplicate requests issued | 10,000 |
| Additional underlying unlearning calls | 0 |
| Median duplicate lookup latency | $7.96\,\mu$s |
| 95th-percentile lookup latency | $8.35\,\mu$s |
| Conflicting key/payload reuse rejected | Yes |
| Simulated pre-commit crash rolled back | Yes |
| Retry after crash executes once | Yes |

interval uses 5,000 hierarchical bootstrap samples, and the reported permutation $p$-value uses 5,000 run-label permutations. All JS divergences are computed from softmax predictive distributions with natural logarithms.