RETENTION IS LEAKAGE: FORGETTING DOES NOT CONFER PRIVACY IN FEDERATED CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated continual learning is often presented as private by design. Clients keep their raw data, and the data of an early task is assumed to fade from the system as the model forgets it. We test this assumption against an adversary who observes every broadcast rather than only the final model. The assumption fails in two ways. First, forgetting does not remove leakage. On CIFAR-100 and ImageNet-R, six further tasks of plain federated averaging erase about three quarters of the above-chance accuracy on an early task. The transcript still holds 87% and 97% of the above-chance membership leakage about that task. Second, retention is leakage. Every anti-forgetting mechanism we test preserves most of the accuracy and keeps or amplifies leakage. Exemplar replay raises above-chance membership leakage by a factor of 2.6 to 4.7 over six tasks. The factor grows with the share of client data that the buffer stores. A method that retains exact second-moment statistics exposes members with a true positive rate of at least 0.92 at a 1% false positive rate. Secure aggregation does not change this. We explain both failures with a lifelong accounting framework. It names the unit of privacy, records which releases read each datum, and separates retention operators that read old data again from operators that do not. Only the second kind admits a privacy cost that is independent of the length of the stream. We build a retention operator of that kind, which retains the release rather than the data. Under example-level privacy at $\epsilon = 4$ it reaches 81% accuracy on CIFAR-100, against 88% without privacy, and a likelihood-ratio audit finds no leakage above chance.

1 INTRODUCTION

Federated continual learning (FCL) enables collaborative training of models across multiple clients whose data arrive over time, without sharing private raw data or forgetting previously learned lessons (Yoon et al., 2021; Dong et al., 2022a). Two fundamental challenges appear together. First, the client data are not identically distributed. Therefore, the aggregated model drifts away from any single client (McMahan et al., 2017; Kairouz et al., 2021b). Second, tasks arrive in sequence. Therefore, models can forget what they learned earlier (Kirkpatrick et al., 2017; Rebuffi et al., 2017). A method must address both challenges in a single framework.

The core motivation of FCL is to combine data privacy with continual learning, although FCL does not by itself guarantee privacy. Almost every work opens with the fundamental argument of conserving raw data and sharing only the model updates. In FCL, this argument is continually extended along the time axis. Hence, over a sequence of tasks, once the reaches the last of the sequence, it is assumed the data of the first task of the sequence is no longer exposed. The paper demonstrates that this extension does not hold. What the model forgets, the transcript still holds.

Let us consider how a conventional FCL framework works. Typically, FCL heavily converges on frozen, pretrained backbones with small trainable modules as a core architecture paradigm (Dosovitskiy et al., 2021; Wang et al., 2022b;a). A client no longer sends a gradient over millions of parameters. Rather, it transmits a compact summary of its own history. Specifically, prototype methods such as Prototypes-injected Prompt (PIP) (Ma'sum et al., 2024), Federated Prototype-Augmented Prompt Learning (FPPL) (He et al., 2024), etc., send per-class feature means and the counts that weight them (Guo et al., 2024). Prompt methods Fed-CPrompt (Bagwe et al., 2023), C^2Prompt (Xu et al., 2025), Powder (Piao et al., 2024), etc., send a pool of learned keys and values, and in recent work the local class distribution that selects them. In addition, adapter methods such as MoAFCL (Zhang & Liu, 2025), DOLFIN Moussadek et al. (2025), FedCAF (Peng et al., 2024),

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.

etc., send low-rank factors (Guo et al., 2024; 2026). Replay methods such as generative replay (Qi et al., 2023) and diffusion-driven replay (Liang et al., 2024) send generated samples or keep latent exemplars that shape every later update (Zhang et al., 2023; Khademi Nori et al., 2025). Closed-form methods LoRM (Salami et al., 2025), FedRAN (Arockiaraj et al., 2026), and AFCL (Shenaj et al., 2023) send second-moment statistics (Tang et al., 2026). All these approaches share one property, i.e., each is built to persist. A prototype is useful precisely because it still describes task one after task ten has been learned. Persistence is the anti-forgetting mechanism. It is also the reason privacy does not decay. The quantity that protects accuracy on an old task is the quantity that preserves information about the data that task came from. In FCL, we state the claim that retention and leakage are two readings of the same thing.

Contributions. This paper makes four primary contributions.

(1) A vocabulary that makes the question answerable. We group releases into eight artifact families. We require every privacy claim to name its unit, and every method to declare which examples each release read (Section 3).

(2) A measurement of what persists. We audit membership leakage over elapsed tasks with likelihood-ratio tests (Carlini et al., 2022a), on the transcript, the aggregate left by secure aggregation (Bonawitz et al., 2017), the retained state, and the final checkpoint (Section 4). Across seven methods and three benchmarks, forgetting does not remove leakage, and every anti-forgetting mechanism keeps or raises it (Section 6).

(3) A test for when lifelong privacy is possible. A retention operator is contractive when each datum influences a bounded number of releases. Contractive operators admit a privacy cost that does not grow with the number of tasks. Accumulating operators with additive Gaussian noise admit no finite lifelong guarantee. Exemplar replay is accumulating even for a single example.

(4) An operator on the right side of that line. Our operator retains the release rather than the data. Its example-level guarantee is fixed for any number of tasks. At $\epsilon = 4$ it reaches 81% accuracy on CIFAR-100, against 88% without privacy, and the audit finds no leakage above chance.

2 Related work

Federated continual learners differ mainly in what they retain. They keep task-specific parameters (Yoon et al., 2021), client exemplar buffers with distillation (Dong et al., 2022a; Khademi Nori et al., 2025), synthetic replay (Zhang et al., 2023), orthogonal subspaces (Bakman et al., 2024), prototypes and adapters (Guo et al., 2024), closed-form statistics (Tang et al., 2026), and prompts or experts (Wang et al., 2022b; Xu et al., 2025; Guo et al., 2026). They are evaluated on accuracy and forgetting (Li et al., 2025). To our knowledge, none measures what the retained artifacts reveal about old data as later tasks arrive. Privacy attacks on federated learning read one round or one model (Melis et al., 2019; Nasr et al., 2019; Zhu et al., 2019; Geiping et al., 2020; Boenisch et al., 2023). Time enters only in centralised work. Models forget early examples (Jagielski et al., 2023), removing exposed examples exposes others (Carlini et al., 2022b), and visual prompts leak in a single task (Wu et al., 2024). Differential privacy for streams (Dwork et al., 2010; Chan et al., 2011) and private training (Abadi et al., 2016; McMahan et al., 2018; Kairouz et al., 2021a; Choquette-Choo et al., 2023) account for runs in which each example contributes a known number of times. Our accountant asks whether a retention operator keeps that number bounded. Lai et al. (2022) fix one budget over a growing stream for a single centralised learner. We seek that property in federation (Appendix G gives more detail).

3 Problem setting and threat model

3.1 The release channel and the adversary

We consider a federation of a set of clients $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$, where $n \in \mathbb{Z}_{>0}$. We define tasks as $T = \{t_k, t_{k+1}, \ldots\}$ where $k, k+1, \ldots$ denotes the infinite monotonically increasing sequence that tasks arrive in. Each client $c_j \in \mathcal{C}$ has private data $D_{c_j}^{t_k}$ during the execution of the task t_k that span a set of rounds $R_k = \{r_1, r_2, \cdots, r_m\}$ where $j \in [1, n]$. In each round $r_i \in R_k$, each participating

client $c_j \in \mathcal{C}_{r_i}$ computes an artifact $a_{c_j}^{r_i}$ from its data and the current global state, where $\mathcal{C}_{r_i} \subseteq \mathcal{C}$ are the clients that participate in round $r_i \in R_k$. The server aggregates all computed artifacts across participating clients in the round r_i into a broadcast A^{r_i}. Now, let $D_{\mathcal{C}}^{T}$ denote the collection of all private data in the federation, over every client $c_j \in \mathcal{C}$ and every task $t_k \in T$ across all rounds in R_k. We introduce a *release channel* $\mathcal{M}$, that maps this collection to a transcript $\mathcal{A}$, i.e.,

$$\mathcal{M} \;:\; D_{\mathcal{C}}^{T} \;\longmapsto\; \mathcal{A} = (A^{r_i}, A^{r_{i+1}}, A^{r_{i+2}}, \dots). \tag{1}$$

Every generated $\mathcal{A}$ captures all the computed artifacts of participating clients across all rounds.

We model the adversary in our formulation as an honest-but-curious (passive) server or participation client, that follows the protocol and only attempts to learn from what it receives without injecting anything. Unlike the malicious server of Boenisch et al. (2023), it never alters a model or an update to extract data. The adversary observes the channel up to an arbitrary round r_i, which we call the observation time, and denoted as τ. We assume a channel is adaptive, because each artifact depends on the broadcasts before it, gradually increasing to the round τ. This can be factorized as $p(\alpha^{r_1:\tau} \mid \mathcal{D}) = \prod_{r_i \le \tau} p_{r_i}(\alpha^{r_i} \mid \alpha^{r_1:r_{i-1}}, \mathcal{D})$, where α^{r_i} is a realised value of A^{r_i} and $\alpha^{r_1:r_{i-1}}$ is a realised value of the broadcasts before round r_i.

We introduce the retention operator Ret that decides what the server carries forward. After each round r_i, Ret takes the broadcasts of all previous rounds up to r_i, and returns the state that the server keeps for round r_{i+1}. We formally denote this state by Σ^{r_i} , so $\Sigma^{r_i} = \mathrm{Ret}(A^{r_1}, \dots, A^{r_i})$. For an observation time τ we consider four views:

$$\begin{aligned} \mathcal{V}_{\text{client}} &= \{a_{c_j}^{r_i} : c_j \in \mathcal{C}_{r_i},\ r_i \le \tau\}, & \mathcal{V}_{\text{full}} &= \{A^{r_i} : r_i \le \tau\}, \\ \mathcal{V}_{\text{state}} &= \{\Sigma^{r_i} : r_i \le \tau\}, & \mathcal{V}_{\text{final}} &= \{\Sigma^{\tau}\}. \end{aligned} \tag{2}$$

The server sees the uploads $\mathcal{V}_{\text{client}}$ only without secure aggregation. It always sees the transcript $\mathcal{V}_{\text{full}}$, as does any client that participates throughout. The other two views are functions of it.

3.2 The artifact ledger and the unit of privacy

We analyse a method only through what it releases. Each release is recorded as $\rho = (r_i, t_k, c_j, f, a_{c_j}^{r_i}, \omega(\rho))$ in the *artifact ledger* $\mathcal{R}$. The family f is one of eight: model deltas (F_1), prototypes and feature moments (F_2), prompt pools (F_3), low-rank adapter factors (F_4), second-moment statistics (F_5), generative replay parameters (F_6), per-class counts (F_7), and raw exemplars (F_8). The *touched set* $\omega(\rho)$ lists every private datum whose value the release reads, including reads through a local replay buffer. Reading the transcript does not count, because it is already public. A release without a declared touched set is treated as reading all of the client's data.

A differential privacy guarantee has no meaning until it names what it protects (Dwork et al., 2006). Datasets are neighbours under a unit u, written $\mathcal{D} \sim_u \mathcal{D}'$, when one is obtained from the other by adding or removing one instance x of that unit. We write $D(x)$ for the data of x. A mechanism is (ϵ, δ)-DP under u when

$$\Pr[\mathcal{M}(\mathcal{D}) \in E] \;\le\; e^{\epsilon} \Pr[\mathcal{M}(\mathcal{D}') \in E] + \delta \quad \text{for every event } E \text{ and every } \mathcal{D} \sim_u \mathcal{D}'. \tag{3}$$

We use five units (Appendix Table 5): one example (U_1), one client in one task (U_2), one client in a window of W tasks (U_3), one client over the whole stream (U_4), and one person (U_5). In our benchmarks each person contributes one record, so U_5 equals U_1. Guarantees under different units are not comparable, so every claim in this paper carries its unit.

3.3 Influence horizon and two accounting results

Definition 1 (Influence horizon)**.** Fix a unit u and an instance x. The influence set $I_\tau(x)$ is the set of rounds $r_i \le \tau$ at which $p_{r_i}(\cdot \mid \alpha^{r_1:r_{i-1}}, \mathcal{D}) \neq p_{r_i}(\cdot \mid \alpha^{r_1:r_{i-1}}, \mathcal{D}')$ for some prefix and some $\mathcal{D} \sim_u \mathcal{D}'$ that differ in x. The influence horizon is $h(\tau) = \sup_x |I_\tau(x)|$. A retention operator is *contractive* under u when $h = \sup_\tau h(\tau) < \infty$, and *accumulating* when $h(\tau) \to \infty$.

A statistic computed once per task and never read again is contractive under U_1 and U_2. Training on a replay buffer is accumulating under U_1, because every later task reads the stored examples again. Any method that keeps training on a client's new data is accumulating under U_4.

Theorem 2 (Bounded influence gives stream-independent privacy)**.** *Fix a unit u. Suppose that (A1) for every round r_i and every prefix, the conditional release $p_{r_i}(\cdot \mid \alpha^{r_1:r_{i-1}}, \cdot)$ is (ϵ_0, δ_0)-DP under u, and (A2) the operator is contractive with horizon $h < \infty$. Then for every τ the transcript $(A^{r_1}, \ldots, A^{\tau})$ is $(h\epsilon_0, h\delta_0)$-DP under u, and so is the unbounded transcript $\mathcal{A}$. The bound does not depend on τ.*

Proposition 3 (Accumulating operators diverge)**.** *Fix a unit u and $\delta \in (0, 1/2)$. Suppose that at each round $r_i \in I_\tau(x)$ the mechanism releases $g_{r_i}(\mathcal{D}) + \xi^{r_i}$, where g_{r_i} does not depend on the prefix and $\xi^{r_i} \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$ is independent across rounds. Suppose there exist $\mathcal{D} \sim_u \mathcal{D}'$ differing in x with $\|g_{r_i}(\mathcal{D}) - g_{r_i}(\mathcal{D}')\|_2 \geq \Delta_0 > 0$ for every $r_i \in I_\tau(x)$, and write $m(\tau) = |I_\tau(x)|$. Once $m(\tau) \geq 4\sigma^2/\Delta_0^2$, the smallest ϵ for which the transcript is (ϵ, δ)-DP under u satisfies*

$$\epsilon(\tau) \;\geq\; \frac{\Delta_0^2\, m(\tau)}{8\sigma^2} \;-\; c_\delta, \qquad c_\delta = -\tfrac{1}{2}\log(2\pi) - \log\big(\Phi(1) - \delta\big), \tag{4}$$

where Φ is the standard normal distribution function. If the operator is accumulating, then $\epsilon(\tau) \to \infty$ and no finite lifelong guarantee exists.

Appendices A.1 and A.2 give the proofs and extend Proposition 3 to adaptive releases. Together the two results give our design rule: a bounded influence horizon decides whether a lifelong guarantee exists.

3.4 What secure aggregation removes

Secure aggregation is the standard answer to leakage in federated learning (Bonawitz et al., 2017). Its ideal form reveals $A^{r_i} = \sum_{c_j \in \mathcal{C}_{r_i}} a_{c_j}^{r_i}$ and nothing else.

Proposition 4 (What secure aggregation removes)**.** *Assume ideal secure aggregation. (i) Every attack on $\mathcal{V}_{\text{full}}$, $\mathcal{V}_{\text{state}}$ or $\mathcal{V}_{\text{final}}$ has the same success rate with and without it. (ii) If uploads are sums, $a_{c_j}^{r_i} = \sum_{z \in D_{c_j}^{t_k}} \phi(z)$, adding an example z changes the aggregate by exactly $\phi(z)$. (iii) Let uploads be class means $\mu_{c_j,y}$ over the $n_{c_j,y}$ examples of class y at client c_j, aggregated into the count-weighted mean μ_y over all n_y examples of that class. Adding an example z of class y moves $\mu_{c_j,y}$ by $(z - \mu_{c_j,y})/(n_{c_j,y} + 1)$ and moves μ_y by $(z - \mu_y)/(n_y + 1)$.*

Appendix A.3 gives the proof for each part.

4 Measuring leakage and bounding it

4.1 Lifelong accounting and the transcript audit

Lifelong accounting. The Lifelong accounting phase computes the privacy cost of a method from its ledger. It trains the FCL method on the task stream and, at every round, stores each client's release as a ledger record with its touched set (Section 3.2). After each task t_k, it counts, for every instance x of the unit u, the records up to task t_k whose touched set contains a datum of $D(x)$. Let m_k be the largest count over all instances. When touched sets are declared honestly, m_k bounds the influence horizon (Lemma 10, Appendix A.4). If every release were a Gaussian mechanism with noise multiplier z, the transcript up to task t_k would be $(\epsilon_u(t_k), \delta)$-DP under u with

$$\epsilon_u(t_k) \;=\; \min_{\alpha > 1}\Big[\frac{m_k\,\alpha}{2z^2} + \frac{\log(1/\delta)}{\alpha - 1}\Big] \;=\; \frac{m_k}{2z^2} + \frac{\sqrt{2\,m_k \log(1/\delta)}}{z} \tag{5}$$

(Proposition 11, proved in Appendix A.4). The cost is constant when m_k is bounded and grows linearly when it is not. For the non-private methods we audit, it prices the pattern of data access, not a guarantee. The full algorithm for the lifelong accounting with its detailed description is presented in Appendix B.1

Transcript audit. The Transcript audit phase measures membership leakage with the likelihood Ratio Attack (LiRA), a membership inference test Carlini et al. (2022a). It trains the method on N shadow federations. In each shadow, every training record is included independently with probability $1/2$. At every round, each target record receives a score computed from the released artifact: the logit

margin for model updates, the negative distance to its class prototype for prototypes, and the leverage $x^\top R^{-1}x$ for a second-moment matrix R. On calibration shadows, the test fits one Gaussian to the scores of shadows that contain the target and one to those that do not. On evaluation shadows, it sums the per-round log-likelihood ratios over the rounds that the adversary reads. The *transcript* adversary reads every round from the start of the target's task t_k to the end of task t_{k+e}. The *checkpoint* adversary reads only the last of these rounds. We report the true positive rate at a false positive rate of $\beta = 1\%$, written TPR.

We compare leakage with accuracy on the same target tasks $\mathcal{K}$, fixed in advance, so the tested population never changes. Let $M_{k,k'}$ be the accuracy on task $t_{k'}$ after task t_k, and let $a_0(t_k)$ be the accuracy of a uniform guess among the classes seen after task t_k. We remove chance and normalise at the end of the target task:

$$A(e) = \frac{\sum_{t_k \in \mathcal{K}} \big(M_{k+e,\,k} - a_0(t_{k+e})\big)}{\sum_{t_k \in \mathcal{K}} \big(M_{k,k} - a_0(t_k)\big)}, \qquad L(e) = \frac{\mathrm{TPR}(e) - \beta}{\mathrm{TPR}(0) - \beta}. \tag{6}$$

Both curves start at 1. A value above 1 means the signal has grown since the task ended. We present the complete algorithm for the transcript audit with its detailed description in Appendix B.2

Definition 5 (Half-life and retention at horizon)**.** The half-life h_ψ of $\psi \in \{A, L\}$ is its first crossing of $1/2$, with linear interpolation between tasks. If no crossing occurs by the horizon E, the half-life is censored and we report $h_\psi > E$. The decoupling ratio is $\varrho = h_L/h_A$. The retention at horizon is the pair $\big(A(E), L(E)\big)$.

A weak attack does not prove that little leaks. Proposition 12 gives the ceiling that no test can exceed. If the transcript is (ϵ, δ)-DP under U_1, every test satisfies $\mathrm{TPR} \le e^\epsilon\,\mathrm{FPR} + \delta$ (Appendix A.5). We use this ceiling to check our operator.

4.2 A RETENTION OPERATOR WITH BOUNDED INFLUENCE

Accumulating methods share one cause. Their retention operators read old data again. Our operator follows one principle: *retain the release, not the data*. It summarises each task once, adds noise, and releases the summary. Everything it carries forward is a function of past releases. Carrying a release forward is post-processing, so retention costs no privacy.

We instantiate the principle with ridge regression over frozen features. Let $P \in \mathbb{R}^{d\times q}$ be an orthonormal projection fitted on public reference data. Each feature x is projected and clipped, and each label y is one-hot:

$$\bar{x} = P^\top x \cdot \min\big(1,\, B/\|P^\top x\|_2\big), \qquad G_{c_j}^{t_k} = \sum_{(x,y)\in D_{c_j}^{t_k}} \bar{x}\bar{x}^\top, \qquad H_{c_j}^{t_k} = \sum_{(x,y)\in D_{c_j}^{t_k}} \bar{x}\, y^\top. \tag{7}$$

Client c_j stacks both statistics into $S_{c_j}^{t_k}$. Under U_2 it also clips $S_{c_j}^{t_k}$ to norm C. Our operator uses one round per task, so task t_k spans the single round r_k. The release of task t_k and the retained state are

$$\begin{aligned} A^{r_k} &= \textstyle\sum_{c_j \in \mathcal{C}_{r_k}} S_{c_j}^{t_k} + \xi^{r_k}, & \tilde{R}^{t_k} &= \gamma \tilde{R}^{t_{k-1}} + \tilde{G}^{t_k}, \\ \tilde{Q}^{t_k} &= \gamma \tilde{Q}^{t_{k-1}} + \tilde{H}^{t_k}, & \tilde{W}^{t_k} &= \big(\Pi_+(\tilde{R}^{t_k}) + \lambda \mathbb{I}\big)^{-1} \tilde{Q}^{t_k}, \end{aligned} \tag{8}$$

where $\xi^{r_k} \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$, $(\tilde{G}^{t_k}, \tilde{H}^{t_k})$ is A^{r_k} reshaped with $\tilde{G}^{t_k}$ symmetrized, Π_+ projects onto positive semidefinite matrices, $\gamma \in [0, 1]$ is a decay, and $\lambda > 0$ is a ridge parameter. The summary of this process is presented in Algorithm 1 with the detailed description in Appendix B.3.

The sensitivity Δ_u is at most $\sqrt{B^4 + B^2}$ under U_1 and at most C under U_2 (Lemma 13, Appendix A.6).

Theorem 6 (Retaining the release is free)**.** *Let $u \in \{U_1, U_2\}$ and let σ be calibrated to Δ_u so that one Gaussian release is (ϵ_0, δ_0)-DP. Then the transcript $(A^{r_1}, A^{r_2}, \dots)$ of Algorithm 1 is (ϵ_0, δ_0)-DP under u over the whole unbounded stream. The classifiers $(\tilde{W}^{t_1}, \tilde{W}^{t_2}, \dots)$ satisfy the same guarantee, for every decay $\gamma \in [0, 1]$ and every ridge parameter $\lambda > 0$.*

Corollary 7 (Other units)**.** *Calibrate σ to Δ_{U_2}. Under U_3 with a window of W tasks, the transcript of Algorithm 1 is $(W\epsilon_0, W\delta_0)$-DP. Under U_4, a client that contributes to infinitely many tasks has no finite lifelong guarantee, for this or any mechanism covered by Proposition 3.*

Algorithm 1 Bounded-influence retention: retain the release, not the data

Require: public projection P; feature clip B; client clip C (unit U_2 only); unit u; target (ϵ_0, δ_0); decay γ; ridge λ

$\sigma \leftarrow \sigma_{\mathrm{AG}}(\epsilon_0, \delta_0, \Delta_u)$; $\tilde{R} \leftarrow 0$; $\tilde{Q} \leftarrow 0$
for $k = 1, 2, \ldots$ **do**
  for $c_j \in \mathcal{C}_{r_k}$ **do**
    $S_{c_j}^{t_k} \leftarrow (G_{c_j}^{t_k}, H_{c_j}^{t_k})$ by Eq. 7; if $u = U_2$, clip $S_{c_j}^{t_k}$ to norm C
    declare $\omega \leftarrow$ identifiers of $D_{c_j}^{t_k}$
  end for
  release $A^{r_k} \leftarrow \sum_{c_j \in \mathcal{C}_{r_k}} S_{c_j}^{t_k} + \xi^{r_k}$ with $\xi^{r_k} \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$
  reshape A^{r_k} into $(\tilde{G}, \tilde{H})$; $\tilde{G} \leftarrow (\tilde{G} + \tilde{G}^\top)/2$
  $\tilde{R} \leftarrow \gamma \tilde{R} + \tilde{G}$; $\tilde{Q} \leftarrow \gamma \tilde{Q} + \tilde{H}$; $\tilde{W}^{t_k} \leftarrow \left(\Pi_+(\tilde{R}) + \lambda \mathbb{I}\right)^{-1} \tilde{Q}$
  discard $D_{c_j}^{t_k}$ and every $S_{c_j}^{t_k}$
end for

Appendix A.6 proves both results. The key step is that each release reads the data of one task only, so Theorem 2 applies with $h = 1$. Retention strength and privacy are therefore decoupled. For $\gamma < 1$ the error of $\tilde{W}^{t_k}$ also stays bounded as k grows (Proposition 14, Appendix A.7). For a client that keeps contributing, the honest promise is U_2 or U_3, stated as such.

5 Experimental setup

Data and stream. We use CIFAR-100 (Krizhevsky, 2009), CUB-200-2011 (Wah et al., 2011) and ImageNet-R (Hendrycks et al., 2021) (Appendix Table 6). Each image is mapped once to the 768-dimensional class token of a frozen ViT-B/16 pretrained on ImageNet-21k (Dosovitskiy et al., 2021). Each dataset forms ten tasks with disjoint classes. Ten clients take part in every task, and the examples of each class are divided among them by a Dirichlet draw with parameter 0.5. Each task has one round of federated averaging (McMahan et al., 2017), in which every client runs 30 epochs of full-batch gradient descent on a linear classifier. A public reference split, disjoint from all clients, is used only to fit and tune our operator.

Methods. We reimplement seven federated continual learners over the same features (Appendix Table 7). They are FedAvg (M0), a GLFC-style buffer with distillation (M1) (Dong et al., 2022a), Gaussian replay (M2) (Zhang et al., 2023), orthogonal projection (M3) (Bakman et al., 2024), prototypes (M4) (Guo et al., 2024), exemplar replay (M5) (Khademi Nori et al., 2025), and a closed-form analytic method (M8) (Tang et al., 2026). They cover five of the eight families. Each keeps the retention mechanism of its source and drops the parts that need a trainable backbone, so its accuracy is not comparable to published numbers.

Audit and accounting. The target tasks are $\mathcal{K} = \{t_1, t_2, t_3, t_4\}$ and the horizon is $E = 6$. Five records from each client shard of each target task are targets. We train 1,024 shadow federations per seed over three seeds. M0 uses 4,096 shadows and five seeds (Appendix D). Of the shadows, 80% fit the test and 20% evaluate it. Every interval is a 95% hierarchical bootstrap interval from 2,000 resamples. Accuracy averages five seeds. The accountant uses $z = 2$, $\delta = 10^{-5}$ and a window of $W = 3$ tasks. At this noise a single release costs $\epsilon = 2.53$.

Our operator. We clip features to $B = 1$ and select q, λ and C on the public reference set (Appendix Table 9). We run $\epsilon_0 \in \{0.5, 1, 2, 4, 8, \infty\}$ with $\delta_0 = 10^{-5}$, units U_1 and U_2, decay $\gamma \in \{1, 0.9, 0.7\}$, and three seeds. We audit it on CIFAR-100 under U_1 with $\gamma = 1$ and $\epsilon_0 \in \{1, 4, \infty\}$, using 1,024 shadows that each draw their own noise.

6 Results

Forgetting does not remove leakage from the transcript: Plain federated averaging (M0) has no mechanism against forgetting, so it directly tests whether forgetting protects old data (Table 1).

FedAvg forgets quickly. On CIFAR-100 and ImageNet-R its above-chance accuracy on a task halves within one further task, after 0.87 and 0.76 tasks. Leakage does not follow. The transcript attacker

Table 1: Plain federated averaging (M0). A random guess scores a TPR of 0.01. A ratio marked $\geq$ is a lower bound, because the leakage half-life exceeds the six observed tasks. Brackets give 95% intervals.

	CIFAR-100	CUB-200-2011	ImageNet-R
Accuracy half-life h_A (tasks)	0.87 [0.80, 0.97]	2.22 [1.66, 2.95]	0.76 [0.72, 0.80]
Leakage half-life h_L (tasks)	> 6	1.08 [0.88, 1.96]	> 6
Decoupling ratio ϱ	≥ 6.87 [6.19, 7.49]	0.49 [0.35, 1.02]	≥ 7.91 [7.49, 8.31]
TPR at $e = 0$	0.028 [0.024, 0.031]	0.058 [0.054, 0.063]	0.074 [0.069, 0.080]
TPR at $e = 6$, transcript	0.026 [0.022, 0.030]	0.025 [0.023, 0.029]	0.072 [0.062, 0.081]
TPR at $e = 6$, checkpoint	0.019 [0.017, 0.022]	0.024 [0.021, 0.027]	0.047 [0.041, 0.053]

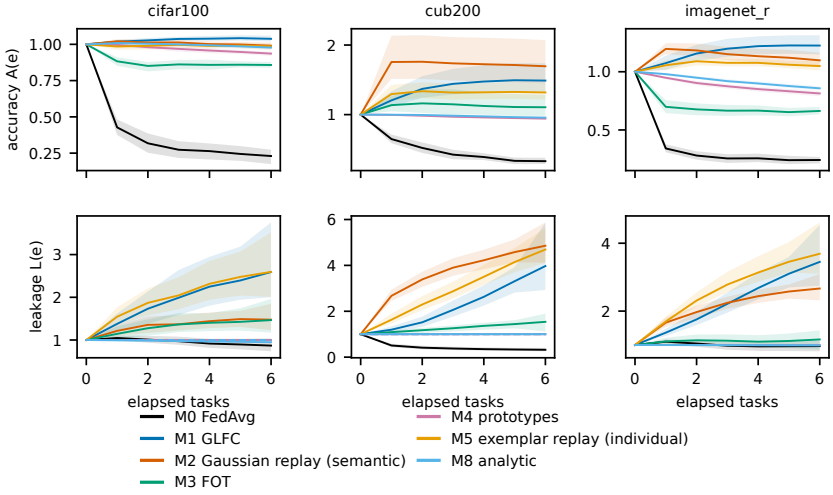


Figure 1: Accuracy $A(e)$ (top) and transcript leakage $L(e)$ (bottom) over e elapsed tasks, relative to their values when the task ended. Dashed lines mark per-client releases, which never change. Bands are 95% intervals.

starts at a TPR of 0.028 and 0.074 and still reaches 0.026 and 0.072 six tasks later. These values keep 87% and 97% of the above-chance leakage (Table 2), and the leakage half-life exceeds the observed horizon. Leakage therefore outlives accuracy by a factor of at least 6.9 on CIFAR-100 and 7.9 on ImageNet-R.

The checkpoint attacker, which reads only the latest model, falls to a TPR of 0.019 and 0.047, and on CIFAR-100 its leakage halves after 1.91 tasks, as the centralised finding of Jagielski et al. (2023) predicts. The gap between the two attackers is what the transcript retains. What the latest model forgets, the earlier broadcasts still hold.

CUB-200-2011 is the exception. There both attackers fall to about 0.025, and leakage decays faster than accuracy. The ratio of half-lives is 0.49, with an interval from 0.35 to 1.02. Forgetting can reduce leakage. It does not do so reliably.

Every retention mechanism keeps or increases leakage The six methods that resist forgetting do their job. On CIFAR-100 they keep 86% to 104% of their above-chance accuracy after six tasks, and at least 66% on every dataset (Figure 1 and Table 2). None of them loses leakage in the process. Their leakage after six tasks is 0.95 to 4.86 times its initial value.

Replay amplifies leakage the most. The exemplar replay method M5 raises it by factors of 2.59, 3.69 and 4.69 on CIFAR-100, ImageNet-R and CUB-200-2011. Gaussian replay (M2) stores no examples but replays samples drawn from their statistics. Its leakage grows by 1.48, 2.67 and 4.86. The projection method M3 reads no old data, yet its leakage also grows, by 1.47 and 1.54 on CIFAR-100 and CUB-200-2011. We did not isolate the cause of this growth.

The growth follows how much data the buffers hold. With at most 10 examples per class per client, buffers ever hold 19% of the training data on CIFAR-100, 46% on ImageNet-R and 96% on CUB-200-2011. The leakage growth of M5 follows the same order. So does its backward transfer, the change in accuracy on old tasks after later training. It is -0.040, $+0.066$ and $+0.321$ on the three

Table 2: Accuracy and leakage kept six tasks after the target task. A value of $L(6)$ above 1 means leakage grew. For M4 and M8 we show the view left by secure aggregation, and for M8 also the retained state. Brackets give 95% intervals.

	CIFAR-100		CUB-200-2011		ImageNet-R	
Method	$A(6)$	$L(6)$	$A(6)$	$L(6)$	$A(6)$	$L(6)$
M0 FedAvg	0.23 [0.17, 0.27]	0.87 [0.73, 1.04]	0.33 [0.29, 0.38]	0.32 [0.25, 0.40]	0.24 [0.21, 0.27]	0.97 [0.81, 1.12]
M1 GLFC-style	1.04 [1.02, 1.06]	2.59 [2.00, 3.75]	1.49 [1.31, 1.74]	3.97 [2.92, 5.86]	1.23 [1.15, 1.31]	3.45 [2.75, 4.54]
M2 Gaussian replay	0.99 [0.97, 1.01]	1.48 [1.23, 1.85]	1.70 [1.45, 2.07]	4.86 [4.12, 5.87]	1.10 [1.04, 1.16]	2.67 [2.32, 3.09]
M3 FOT	0.86 [0.84, 0.88]	1.47 [1.17, 1.96]	1.11 [0.98, 1.28]	1.54 [1.11, 1.90]	0.66 [0.63, 0.70]	1.16 [0.86, 1.43]
M5 Exemplar replay	0.99 [0.96, 1.00]	2.59 [2.01, 3.52]	1.32 [1.22, 1.46]	4.69 [4.11, 5.63]	1.05 [1.01, 1.09]	3.69 [3.10, 4.61]
M4 Prototypes, aggregate	0.94 [0.92, 0.95]	1.00 [1.00, 1.00]	0.94 [0.93, 0.95]	1.00 [1.00, 1.00]	0.81 [0.79, 0.83]	1.00 [1.00, 1.00]
M8 Analytic, aggregate	0.98 [0.96, 0.99]	1.00 [1.00, 1.00]	0.96 [0.94, 0.97]	1.00 [1.00, 1.00]	0.86 [0.84, 0.88]	1.00 [1.00, 1.00]
M8 Analytic, state	0.98 [0.96, 0.99]	0.95 [0.93, 0.97]	0.96 [0.94, 0.97]	1.00 [1.00, 1.00]	0.86 [0.84, 0.88]	0.99 [0.98, 1.00]

Table 3: TPR six tasks after the target task, by view. Secure aggregation hides the per-client release and leaves the aggregate and the retained state visible. Brackets give 95% intervals.

Method	View	CIFAR-100	CUB-200-2011	ImageNet-R
M4 Prototypes	per-client release	0.343 [0.314, 0.374]	0.996 [0.993, 0.998]	0.732 [0.702, 0.763]
	aggregate	0.058 [0.051, 0.066]	0.832 [0.808, 0.858]	0.240 [0.216, 0.268]
	retained state	0.058 [0.051, 0.066]	0.832 [0.808, 0.858]	0.240 [0.216, 0.268]
M8 Analytic	per-client release	1.000 [1.000, 1.000]	1.000 [1.000, 1.000]	1.000 [1.000, 1.000]
	aggregate	0.999 [0.998, 1.000]	1.000 [1.000, 1.000]	1.000 [1.000, 1.000]
	retained state, all rounds	0.917 [0.899, 0.931]	1.000 [1.000, 1.000]	0.984 [0.978, 0.990]
	retained state, final round	0.701 [0.668, 0.728]	1.000 [0.999, 1.000]	0.906 [0.889, 0.924]

datasets (Appendix Table 10). A positive value means old tasks keep improving after their task ends. With one round per task, these methods keep training on stored data long after the task is over, and leakage rises with accuracy. For the same reason, seven of the nine replay configurations fail our check against FedAvg (Appendix C), and their growth on CUB-200-2011 and ImageNet-R may be larger than under a protocol that learns each task fully. On CIFAR-100, M2 and M5 pass the check, and their leakage still grows.

The aggregates of M4 and M8 are released once and never change, so their leakage is exactly constant. The retained state of M8 keeps 95% to 100% of its initial leakage. Attacks on model updates are weaker in absolute terms, with a TPR of 0.038 to 0.329 six tasks after the target task (Appendix Table 11). The relative growth is the finding. For the four retention methods based on model updates, it exceeds 1 in all 12 cases, and its interval excludes 1 in 11 of them.

The amount retained controls the leakage: To exclude other differences between methods, we vary the amount retained inside one method on CIFAR-100. We set the buffer of M5 and the replay of M2 to 1, 2, 5, 10, 20 and 50 samples per class (Figure 2; values in Appendix Table 12). For both methods, retaining more lowers forgetting and raises leakage at every step. Growing the buffer of M5 from 1 to 50 lowers forgetting from 0.153 to 0.009 and raises the TPR from 0.038 to 0.083. At comparable forgetting, stored examples leak more than replayed statistics. With 10 samples per class, M5 forgets 0.043 at a TPR of 0.062, while M2 forgets 0.049 at a TPR of 0.037.

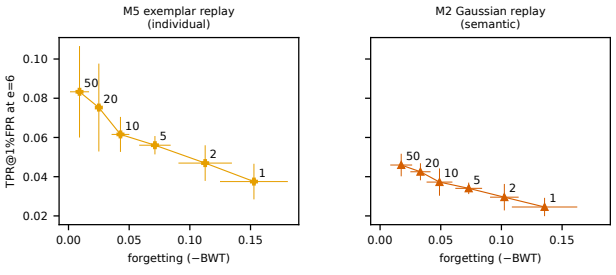


Figure 2: Dose response on CIFAR-100: forgetting ($-$BWT) against TPR six tasks after the target task, as samples per class grow (labels). Bars: 95% intervals, three seeds.

Table 4: Our operator. Left: final accuracy after ten tasks with $\gamma = 1$ and non-private references (intervals in Appendix Table 14). Right: audit on CIFAR-100 under U_1, with the upper end of the 95% TPR interval, the ceiling $e^{\epsilon_0} \cdot 0.01 + \delta_0$, and the empirical lower bound $\hat{\epsilon}$.

	CIFAR-100		CUB-200		ImageNet-R	
ϵ_0	U_1	U_2	U_1	U_2	U_1	U_2
1	0.596	0.015	0.017	0.004	0.082	0.005
4	0.810	0.016	0.088	0.006	0.324	0.006
8	0.818	0.017	0.161	0.007	0.385	0.006
∞	0.884	0.880	0.868	0.848	0.634	0.624
M8, no privacy	0.883		0.871		0.635	
M0, no privacy	0.333		0.386		0.248	

ϵ_0	TPR	Upper	Ceiling	AUC	$\hat{\epsilon}$
1	0.0095	0.0104	0.0272	0.500	0.00
4	0.0087	0.0095	0.5460	0.499	0.00
∞	0.8711	0.8740	1	0.993	9.06

What secure aggregation removes in practice: Table 3 tests the predictions of Proposition 4. For model updates, secure aggregation changes nothing, because the attack reads the broadcast model. For prototypes it helps by an amount that depends on the data. It lowers the TPR of M4 from 0.343 to 0.058 on CIFAR-100, from 0.732 to 0.240 on ImageNet-R, and only from 0.996 to 0.832 on CUB-200-2011. Part (iii) of the proposition explains the order. The datasets have about 400, 120 and 25 training examples per class, and the fewer examples a class has, the more each one shows. For second-moment statistics, secure aggregation does not help, as part (ii) predicts. The aggregate of M8 keeps a TPR of 0.999 or above. Its retained state keeps 0.917 to 1.000 over the transcript, and 0.701 to 1.000 from the final round alone. Retaining exact statistics exposes almost every member, whoever holds them.

Which retention operators admit a lifelong guarantee: We apply the accountant to one CIFAR-100 run of each method (Appendix Table 13). Under U_1 and U_2, M0, M2, M3, M4 and M8 read the data of a task only during that task. Their count stays at 1 or 2, so their cost stays at 2.53 or 3.64 for any number of tasks. M1 and M5 train on their buffers at every later task. Their count grows by about one per task even for a single example, and at 1,000 tasks their example-level cost reaches 201. Under U_4 every method accumulates, because each new task of a client reads that client's data. A bounded count is necessary for a lifelong guarantee, but it is not sufficient. M8 has the smallest count of all, yet it exposes almost every member. A bounded count makes a guarantee affordable. Only noise provides it.

Retaining the release: accuracy and privacy of our operator: Under U_1 our operator trades accuracy for privacy smoothly on CIFAR-100 (Table 4, left; Appendix D covers U_3, U_4 and γ). It reaches 0.596 at $\epsilon_0 = 1$ and 0.810 at $\epsilon_0 = 4$, against 0.884 without noise, which matches the non-private analytic method M8 at 0.883. The cost is larger on the fine-grained datasets. At $\epsilon_0 = 8$ it reaches 0.161 on CUB-200-2011 and 0.385 on ImageNet-R, against 0.868 and 0.634 without noise. The statistics are sums over examples while the noise has a fixed size, so fewer examples per class mean a weaker signal. Under U_2 accuracy stays near chance at every finite ϵ_0. A client-level guarantee must hide a client's whole contribution, and with ten clients the aggregate is only a few times larger than one contribution. With 50 or 100 clients, accuracy at $\epsilon_0 = 8$ rises only to 0.042 and 0.048 (Appendix Table 17).

The audit agrees with the guarantee (Table 4, right). At $\epsilon_0 = 1$ and $\epsilon_0 = 4$ the attack reaches a TPR of 0.0095 and 0.0087. Both are at the level of a random guess and far below the ceilings of 0.0272 and 0.5460. The same attack reaches 0.871 when the noise is removed, so it finds leakage where leakage exists. At $\epsilon_0 = 4$ the attack stays far below the ceiling, so the audit cannot tell whether the bound is loose or the attack is weak.

7 Conclusion

Federated continual learning does not become private as it forgets. Against an adversary who keeps every broadcast, forgetting leaves most of the leakage in place, and every mechanism that prevents forgetting keeps or increases it. The cause is the retention operator. Reading old data again costs privacy at every task. Retaining only past releases costs it once. Our operator of the second kind holds a fixed example-level guarantee over any number of tasks.

Ethics statement

This work studies attacks against published methods in order to correct a privacy claim the field relies on. All datasets are public research releases used under their licences, and we attempt no re-identification of any individual. Experiments ran on a cluster whose data policy forbids protected data. We release the audit code together with the defensive mechanism of Section 4.2.

Reproducibility statement

Every number in this paper is produced by a logged run and written to a CSV before it is plotted or tabulated. We release those CSVs, the run manifests that record the code state and configuration behind each one, the dataset manifest with checksums, and the exact commands that regenerate every figure and table. Appendix A contains complete proofs, Appendix B the measurement algorithms, and Appendix C every setting.

7.1 AI use statement

Generative AI Usage: Generative AI tools such as Claude were used for editorial purposes in the preparation of this manuscript (e.g., grammar refinement, stylistic polishing, and clarity improvements). The authors relied on generative AI tools like Claude for assisting with the creation of skeletal code for certain class and function declarations as well as for routine debugging activities for the core programming logic. All AI-assisted content was reviewed, verified, and validated by the authors, including re-running experiments and checking citations.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2016.

Jebacyril Arockiaraj, Dhruv Parikh, Jayashree Adivarahan, Rajgopal Kannan, and Viktor Prasanna. Accurate and resource-efficient federated continual learning, 2026. URL https://arxiv.org/abs/2606.11480.

Gaurav Bagwe, Xiaoyong Yuan, Miao Pan, and Lan Zhang. Fed-cprompt: Contrastive prompt for rehearsal-free federated continual learning, 2023. URL https://arxiv.org/abs/2307.04869.

Yavuz Faruk Bakman, Duygu Nur Yaldiz, Yahya H. Ezzeldin, and Salman Avestimehr. Federated orthogonal training: Mitigating global catastrophic forgetting in continual federated learning. In *International Conference on Learning Representations (ICLR)*, 2024.

Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

Peter Bandi, Oscar Geessink, Quirine Manson, Marcory van Dijk, Maschenka Balkenhol, et al. From detection of individual metastases to classification of lymph node status at the patient level: The CAMELYON17 challenge. *IEEE Transactions on Medical Imaging*, 38(2), 2019.

Franziska Boenisch, Adam Dziedzic, Roei Schuster, Ali Shahin Shamsabadi, Ilia Shumailov, and Nicolas Papernot. When the curious abandon honesty: Federated learning is not private. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2023.

Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *IEEE Symposium on Security and Privacy (S&P)*, 2022a.

Nicholas Carlini, Matthew Jagielski, Chiyuan Zhang, Nicolas Papernot, Andreas Terzis, and Florian Tramèr. The privacy onion effect: Memorization is relative. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022b.

T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Transactions on Information and System Security*, 14(3), 2011.

Christopher A. Choquette-Choo, H. Brendan McMahan, J. Keith Rush, and Abhradeep Thakurta. Multi-epoch matrix factorization mechanisms for private machine learning. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

Sergey Denisov, H. Brendan McMahan, John Rush, Adam Smith, and Abhradeep Guha Thakurta. Improved differential privacy for SGD via optimal private linear operators on adaptive streams. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Jiahua Dong, Lixu Wang, Zhen Fang, Gan Sun, Shichao Xu, Xiao Wang, and Qi Zhu. Federated class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022a.

Jinshuo Dong, Aaron Roth, and Weijie J. Su. Gaussian differential privacy. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(1), 2022b.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference (TCC)*, 2006.

Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, 2010.

Vitaly Feldman and Tijana Zrnic. Individual privacy accounting via a rényi filter. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients: How easy is it to break privacy in federated learning? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Haiyang Guo, Fei Zhu, Wenzhuo Liu, Xu-Yao Zhang, and Cheng-Lin Liu. PILoRA: Prototype guided incremental LoRA for federated class-incremental learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

Tao Guo, Junwei Chen, and Laizhong Cui. Fed-Duet: Dual expert-orchestrated framework for continual federated vision-language learning. In *International Conference on Learning Representations (ICLR)*, 2026.

Yuchen He, Chuyun Shen, Xiangfeng Wang, and Bo Jin. Fppl: An efficient and non-iid robust federated continual learning framework. pp. 3692 – 3701, 2024. doi: 10.1109/BigData62323.2024.10825220. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85218038911&doi=10.1109%2fBigData62323.2024.10825220&partnerID=40&md5=c2970fd977368d755f508857b6ee9560. Cited by: 5.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Matthew Jagielski, Om Thakkar, Florian Tramèr, Daphne Ippolito, Katherine Lee, Nicholas Carlini, Eric Wallace, Shuang Song, Abhradeep Thakurta, Nicolas Papernot, and Chiyuan Zhang. Measuring forgetting of memorized training examples. In *International Conference on Learning Representations (ICLR)*, 2023.

Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. Practical and private (deep) learning without sampling or shuffling. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021a.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1–2), 2021b.

Milad Khademi Nori, Il-Min Kim, and Guanghui Wang. Federated class-incremental learning: A hybrid approach using latent exemplars and data-free techniques to address local and global forgetting. In *International Conference on Learning Representations (ICLR)*, 2025.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13), 2017.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, et al. WILDS: A benchmark of in-the-wild distribution shifts. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

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

Phung Lai, Han Hu Phan, Li Xiong, Khang Tran, My T. Thai, Tong Sun, Franck Dernoncourt, Jiuxiang Gu, Nikolaos Barmpalios, and Rajiv Jain. Lifelong DP: Consistently bounded differential privacy in lifelong machine learning. In *Proceedings of the Conference on Lifelong Learning Agents (CoLLAs), PMLR 199*, 2022.

Yichen Li, Yuying Wang, Jiahua Dong, Haozhao Wang, Yining Qi, Rui Zhang, and Ruixuan Li. Resource-constrained federated continual learning: What does matter? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Jinglin Liang, Jin Zhong, Hanlin Gu, Zhongqi Lu, Xingxing Tang, Gang Dai, Shuangping Huang, Lixin Fan, and Qiang Yang. Diffusion-driven data replay: A novel approach tonbsp;combat forgetting innbsp;federated class continual learning. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part XXX*, pp. 303–319, Berlin, Heidelberg, 2024. Springer-Verlag. ISBN 978-3-031-73403-8. doi: 10.1007/978-3-031-73404-5_18. URL https://doi.org/10.1007/978-3-031-73404-5_18.

Muhammad Anwar Ma'sum, Mahardhika Pratama, Savitha Ramasamy, Lin Liu, Habibullah Habibullah, and Ryszard Kowalczyk. Pip: Prototypes-injected prompt for federated class incremental learning. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, pp. 1670–1679, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704369. doi: 10.1145/3627673.3679794. URL https://doi.org/10.1145/3627673.3679794.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations (ICLR)*, 2018.

Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *IEEE Symposium on Security and Privacy (S&P)*, 2019.

Ilya Mironov. Rényi differential privacy. In *IEEE Computer Security Foundations Symposium (CSF)*, 2017.

Omayma Moussadek, Riccardo Salami, and Simone Calderara. Dolfin: Balancing stability and plasticity in federated continual learning, 2025. URL https://arxiv.org/abs/2510.13567.

Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *IEEE Symposium on Security and Privacy (S&P)*, 2019.

Danni Peng, Yuan Wang, Huazhu Fu, Qingsong Wei, Yong Liu, and Rick Siow Mong Goh. Learning task-specific initialization for effective federated continual fine-tuning of foundation model adapters. In *2024 IEEE Conference on Artificial Intelligence (CAI)*, pp. 811–816, 2024. doi: 10.1109/CAI59869.2024.00153.

Hongming Piao, Yichen Wu, Dapeng Wu, and Ying Wei. Federated continual learning via prompt-based dual knowledge transfer. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 40725–40739. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/piao24a.html.

Daiqing Qi, Handong Zhao, and Sheng Li. Better generative replay for continual federated learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=cRxYWKiTan.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Riccardo Salami, Pietro Buzzega, Matteo Mosconi, Jacopo Bonato, Luigi Sabetta, and Simone Calderara. Closed-form merging of parameter-efficient modules for federated continual learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ROpY0qRUXL.

Giuseppe Serra and Florian Buettner. Federated continual learning goes online: Uncertainty-aware memory management for vision tasks and beyond. In *International Conference on Learning Representations (ICLR)*, 2025.

Donald Shenaj, Marco Toldo, Alberto Rigon, and Pietro Zanuttigh. Asynchronous federated continual learning, 2023. URL https://arxiv.org/abs/2304.03626.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy (S&P)*, 2017.

Thomas Steinke, Milad Nasr, and Matthew Jagielski. Privacy auditing with one (1) training run. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Jianheng Tang, Jingyu He, Kejia Fan, Run He, Jingchao Wang, Anfeng Liu, Houbing Herbert Song, Leye Wang, Zhanxing Zhu, Huiping Zhuang, and Yunhuai Liu. AFCL: Achieving spatio-temporal invariance to data heterogeneity in federated continual learning. In *Findings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR Findings)*, 2026.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. DualPrompt: Complementary prompting for rehearsal-free continual learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022a.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022b.

Yixin Wu, Rui Wen, Michael Backes, Pascal Berrang, Mathias Humbert, Yun Shen, and Yang Zhang. Quantifying privacy risks of prompts in visual prompt learning. In *33rd USENIX Security Symposium*, 2024.

Abudukelimu Wuerkaixi, Sen Cui, Jingfeng Zhang, Kunda Yan, Bo Han, Gang Niu, Lei Fang, Changshui Zhang, and Masashi Sugiyama. Accurate forgetting for heterogeneous federated continual learning. In *International Conference on Learning Representations (ICLR)*, 2024.

Kunlun Xu, Jiangmeng Li, Yibo Feng, Yongsheng Qi, and Jiahuan Zhou. C^{2}Prompt: Class-aware client knowledge interaction for federated continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with weighted inter-client transfer. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

Dian Zhang and Bingyan Liu. Moafcl: Feature-aware mixture-of-adapter for federated continual learning. In *Proceedings of the 2025 International Conference on Multimedia Retrieval*, ICMR '25, pp. 1804–1813, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400718779. doi: 10.1145/3731715.3733394. URL https://doi.org/10.1145/3731715.3733394.

Jie Zhang, Chen Chen, Weiming Zhuang, and Lingjuan Lyu. TARGET: Federated class-continual learning via exemplar-free distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

APPENDIX

A PROOFS

This appendix restates every result of the main text and proves it. It also states and proves the supporting results that the main text cites. Every proof works with the factorisation $p(\alpha^{r_1:\tau} \mid \mathcal{D}) = \prod_{r_i \le \tau} p_{r_i}(\alpha^{r_i} \mid \alpha^{r_1:r_{i-1}}, \mathcal{D})$ of Section 3.1.

A.1 PROOF OF THEOREM 2

Theorem 2 (Bounded influence gives stream-independent privacy)**.** *Fix a unit u. Suppose that (A1) for every round r_i and every prefix, the conditional release $p_{r_i}(\cdot \mid \alpha^{r_1:r_{i-1}}, \cdot)$ is (ϵ_0, δ_0)-DP under u, and (A2) the operator is contractive with horizon $h < \infty$. Then for every τ the transcript $(A^{r_1}, \dots, A^{\tau})$ is $(h\epsilon_0, h\delta_0)$-DP under u, and so is the unbounded transcript $\mathcal{A}$. The bound does not depend on τ.*

Proof. Fix $\mathcal{D} \sim_u \mathcal{D}'$ differing in the unit instance x, and fix τ.

Step 1. Assign privacy parameters round by round. For $r_i \notin I_\tau(x)$, Definition 1 states that the two conditional laws at round r_i are equal for every prefix. Equal distributions satisfy Equation equation 3 with $\epsilon = 0$ and $\delta = 0$. For $r_i \in I_\tau(x)$, condition (A1) gives (ϵ_0, δ_0). Set $(\epsilon_{r_i}, \delta_{r_i}) = (0, 0)$ in the first case and $(\epsilon_{r_i}, \delta_{r_i}) = (\epsilon_0, \delta_0)$ in the second.

Step 2. The transcript $(A^{r_1}, \dots, A^{\tau})$ is the adaptive composition of the conditional mechanisms of the rounds $r_i \le \tau$ in the factorisation. By the basic adaptive composition theorem for (ϵ, δ) differential privacy, it is $\big(\sum_{r_i \le \tau} \epsilon_{r_i},\ \sum_{r_i \le \tau} \delta_{r_i}\big)$-DP.

Step 3. The sums in Step 2 have non-zero terms only for $r_i \in I_\tau(x)$. There are at most h such rounds by (A2). Hence $\sum_{r_i \le \tau} \epsilon_{r_i} \le h\epsilon_0$ and $\sum_{r_i \le \tau} \delta_{r_i} \le h\delta_0$. Since x was arbitrary, the transcript is $(h\epsilon_0, h\delta_0)$-DP under $\sim_u$.

Step 4. The bound in Step 3 contains no occurrence of τ. It remains to pass to the unbounded transcript. Let $\mathcal{G}$ be the collection of events that depend on finitely many coordinates. $\mathcal{G}$ is an algebra that generates the product sigma-algebra, and Equation equation 3 holds on $\mathcal{G}$ by Step 3. Let $\mathcal{H}$ be the collection of events on which Equation equation 3 holds. If $E_n \uparrow E$ with $E_n \in \mathcal{H}$, continuity of both

probability measures from below gives $\Pr[E] = \lim \Pr[E_n] \le \lim\big(e^{\epsilon} \Pr'[E_n] + \delta\big) = e^{\epsilon} \Pr'[E] + \delta$. The same argument with continuity from above covers $E_n \downarrow E$. So $\mathcal{H}$ is a monotone class that contains $\mathcal{G}$. By the monotone class theorem, $\mathcal{H}$ contains the generated sigma-algebra. The unbounded transcript therefore satisfies the same bound. □

Remark 8. Advanced composition sharpens the constant without changing the conclusion. Under the same assumptions, for any $\delta' > 0$ the transcript is $\big(\sqrt{2h \ln(1/\delta')}\, \epsilon_0 + h\epsilon_0(e^{\epsilon_0} - 1),\ h\delta_0 + \delta'\big)$-DP. This bound is also free of τ. The budget is paid once per influenced release, and a contractive operator caps how many of those exist.

A.2 Proof of Proposition 3

Proposition 3 (Accumulating operators diverge)**.** *Fix a unit u and $\delta \in (0, 1/2)$. Suppose that at each round $r_i \in I_\tau(x)$ the mechanism releases $g_{r_i}(\mathcal{D}) + \xi^{r_i}$, where g_{r_i} does not depend on the prefix and $\xi^{r_i} \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$ is independent across rounds. Suppose there exist $\mathcal{D} \sim_u \mathcal{D}'$ differing in x with $\|g_{r_i}(\mathcal{D}) - g_{r_i}(\mathcal{D}')\|_2 \ge \Delta_0 > 0$ for every $r_i \in I_\tau(x)$, and write $m(\tau) = |I_\tau(x)|$. Once $m(\tau) \ge 4\sigma^2/\Delta_0^2$, the smallest ϵ for which the transcript is (ϵ, δ)-DP under u satisfies*

$$\epsilon(\tau) \ \ge \ \frac{\Delta_0^2\, m(\tau)}{8\sigma^2} \ - \ c_\delta, \qquad c_\delta = -\tfrac{1}{2}\log(2\pi) - \log\big(\Phi(1) - \delta\big), \tag{4}$$

where Φ is the standard normal distribution function. If the operator is accumulating, then $\epsilon(\tau) \to \infty$ and no finite lifelong guarantee exists.

Proof. Step 1. Restrict the adversary to the coordinates released at rounds in $I_\tau(x)$. Discarding coordinates is post-processing. So any ϵ that is valid for the full transcript is valid for the restriction. A lower bound on ϵ for the restriction is therefore a lower bound for the transcript.

Step 2. Since g_{r_i} does not depend on the prefix and the noise is independent across rounds, the restricted view is a single Gaussian vector. Under $\mathcal{D}$ it has law $\mathcal{N}(m_{\mathcal{D}}, \sigma^2\mathbb{I})$ and under $\mathcal{D}'$ it has law $\mathcal{N}(m_{\mathcal{D}'}, \sigma^2\mathbb{I})$, where $m_{\mathcal{D}}$ stacks the vectors $g_{r_i}(\mathcal{D})$ for $r_i \in I_\tau(x)$. The squared distance between the means is $\|m_{\mathcal{D}} - m_{\mathcal{D}'}\|_2^2 = \sum_{r_i \in I_\tau(x)} \|g_{r_i}(\mathcal{D}) - g_{r_i}(\mathcal{D}')\|_2^2 \ge m(\tau)\, \Delta_0^2$.

Step 3. Rotate coordinates so that $m_{\mathcal{D}} - m_{\mathcal{D}'}$ lies along the first axis, and rescale by σ. The remaining coordinates have the same law under both datasets and are independent of the first. The problem reduces to one dimension. Under $\mathcal{D}$ the statistic is $Z \sim \mathcal{N}(\mu, 1)$ and under $\mathcal{D}'$ it is $Z \sim \mathcal{N}(0, 1)$, with $\mu = \|m_{\mathcal{D}} - m_{\mathcal{D}'}\|_2/\sigma \ge \mu_\tau$, where $\mu_\tau = \Delta_0\sqrt{m(\tau)}/\sigma$.

Step 4. Apply Equation equation 3 to the event $E = \{Z > \mu/2\}$. Under $\mathcal{D}$ its probability is $\Phi(\mu/2)$. Under $\mathcal{D}'$ it is $\Phi(-\mu/2)$. The guarantee requires $\Phi(\mu/2) \le e^{\epsilon}\, \Phi(-\mu/2) + \delta$. Because $\mu > 0$ and $\delta < 1/2$, we have $\Phi(\mu/2) - \delta > 0$, and rearranging gives $\epsilon \ge f(\mu) := \log\big(\Phi(\mu/2) - \delta\big) - \log \Phi(-\mu/2)$.

Step 5. The function f is increasing in μ, because $\Phi(\mu/2)$ increases and $\Phi(-\mu/2)$ decreases. Since $\mu \ge \mu_\tau$, we obtain the exact bound

$$\epsilon(\tau) \ \ge \ \log\big(\Phi(\mu_\tau/2) - \delta\big) \ - \ \log \Phi(-\mu_\tau/2). \tag{9}$$

Step 6. For $a > 0$ the Gaussian tail satisfies $\Phi(-a) \le \varphi(a)/a$, where φ is the standard normal density. Taking logarithms with $a = \mu_\tau/2$ gives $-\log \Phi(-\mu_\tau/2) \ge \mu_\tau^2/8 + \frac{1}{2}\log(2\pi) + \log(\mu_\tau/2)$. When $\mu_\tau \ge 2$, which is the condition $m(\tau) \ge 4\sigma^2/\Delta_0^2$, we have $\log(\mu_\tau/2) \ge 0$ and $\Phi(\mu_\tau/2) \ge \Phi(1)$. Substituting both into Equation equation 9 gives $\epsilon(\tau) \ge \mu_\tau^2/8 + \frac{1}{2}\log(2\pi) + \log\big(\Phi(1) - \delta\big)$. Replacing μ_τ^2 by $\Delta_0^2 m(\tau)/\sigma^2$ yields Equation equation 4. The constant c_δ is finite because $\Phi(1) - \delta > 0.34$ for $\delta < 1/2$.

Step 7. If the operator is accumulating then $m(\tau) \to \infty$ by Definition 1. The right side of Equation equation 4 then grows without bound, so no finite ϵ is valid for every τ. □

Remark 9. The proposition assumes that g_{r_i} does not depend on the prefix. The adaptive case follows from composition of Gaussian differential privacy (Dong et al., 2022b). Suppose each influenced round shifts the mean by exactly Δ_0 in noise-scaled norm. The adaptive composition then has the same privacy profile as a single Gaussian with separation $\Delta_0\sqrt{m(\tau)}/\sigma$, and Steps 3 to

7 apply unchanged. This covers gradient perturbation with clipping (Abadi et al., 2016; McMahan et al., 2018), where a canary whose gradient exceeds the clipping norm realises the shift exactly. Correlated-noise mechanisms (Kairouz et al., 2021a; Denisov et al., 2022; Choquette-Choo et al., 2023) change the constant but not the growth in the number of influenced releases. A converse for arbitrary mechanisms with an unbounded horizon remains open. We do not claim it.

A.3 Proof of Proposition 4

Proposition 4 (What secure aggregation removes)**.** *Assume ideal secure aggregation. (i) Every attack on $\mathcal{V}_{\text{full}}$, $\mathcal{V}_{\text{state}}$ or $\mathcal{V}_{\text{final}}$ has the same success rate with and without it. (ii) If uploads are sums, $a_{c_j}^{r_i} = \sum_{z \in D_{c_j}^{t_k}} \phi(z)$, adding an example z changes the aggregate by exactly $\phi(z)$. (iii) Let uploads be class means $\mu_{c_j,y}$ over the $n_{c_j,y}$ examples of class y at client c_j, aggregated into the count-weighted mean μ_y over all n_y examples of that class. Adding an example z of class y moves $\mu_{c_j,y}$ by $(z - \mu_{c_j,y})/(n_{c_j,y} + 1)$ and moves μ_y by $(z - \mu_y)/(n_y + 1)$.*

Proof. Step 1 (i). The ideal functionality removes only the individual uploads, which form $\mathcal{V}_{\text{client}}$. The broadcasts A^{r_i} are computed from the same uploads in both protocols, so $\mathcal{V}_{\text{full}}$ has the same law in both. The states $\Sigma^{r_i} = \text{Ret}(A^{r_1}, \ldots, A^{r_i})$ are measurable functions of $\mathcal{V}_{\text{full}}$, so $\mathcal{V}_{\text{state}}$ and $\mathcal{V}_{\text{final}}$ also have the same law in both. An attack is a randomised map from its input view to a decision. The same map applied to inputs with the same law produces decisions with the same law, so every success rate is equal.

Step 2 (ii). Write $D_{c_j}^{t_k\prime} = D_{c_j}^{t_k} \cup \{z\}$. Additivity gives $a_{c_j}^{r_i\prime} - a_{c_j}^{r_i} = \phi(z)$. The uploads of all other clients are unchanged. The aggregate is a sum over clients, so $A^{r_i\prime} - A^{r_i} = a_{c_j}^{r_i\prime} - a_{c_j}^{r_i} = \phi(z)$.

Step 3 (iii). For any mean over ℓ vectors, adding one vector z gives $\mu' = (\ell\mu + z)/(\ell + 1)$, so $\mu' - \mu = (z - \mu)/(\ell + 1)$. Applying this with $\ell = n_{c_j,y}$ gives the change of the client mean. The count-weighted mean of the client means equals the mean over all n_y examples of class y, so applying the same identity with $\ell = n_y$ gives the change of the aggregate. □

In every case secure aggregation removes attribution. It does not remove the information that the aggregate itself carries, and it is not a substitute for a guarantee under a named unit.

A.4 Touched sets and the ledger accountant

Algorithm 2 measures influence through touched sets, while Definition 1 measures it through conditional laws. The next lemma shows that the two agree in the conservative direction whenever touched sets are declared honestly.

Lemma 10 (Touched sets bound influence)**.** *Suppose every record obeys the declaration rule of Section 3.2. Concretely, suppose that a release at round r_i reads a datum in $D(x)$, or reads a stored quantity other than the transcript whose value depends on such a datum. Then $\omega(\rho) \cap D(x) \neq \emptyset$ for some record ρ at round r_i. Let $\hat{I}_\tau(x)$ be the set of rounds $r_i \leq \tau$ that carry such a record. Then $I_\tau(x) \subseteq \hat{I}_\tau(x)$ for every instance x and every round τ. In particular, if τ is the last round of task t_k, then $|I_\tau(x)| \leq m_k(x)$ and $h(\tau) \leq m_k$, with m_k as computed on line 16 of Algorithm 2.*

Proof. Step 1. Take any round $r_i \leq \tau$ with $r_i \notin \hat{I}_\tau(x)$. No record at round r_i has a touched set that meets $D(x)$. By the declaration rule, no computation at round r_i reads $D(x)$ or any stored quantity that depends on it, apart from the transcript.

Step 2. Every upload at round r_i is therefore a measurable function of three inputs: the transcript prefix $\alpha^{r_1:r_{i-1}}$, data outside $D(x)$, and fresh randomness. The broadcast A^{r_i} is a function of these uploads, so it depends on the same three inputs. Datasets $\mathcal{D}$ and $\mathcal{D}'$ that differ only in x agree on the second input by definition, and the third input has the same law under both.

Step 3. For every prefix, the conditional law of A^{r_i} is therefore the same under $\mathcal{D}$ and $\mathcal{D}'$. By Definition 1, $r_i \notin I_\tau(x)$.

Step 4. Steps 1 to 3 show that $r_i \notin \hat{I}_\tau(x)$ implies $r_i \notin I_\tau(x)$, so $I_\tau(x) \subseteq \hat{I}_\tau(x)$. Now let τ be the last round of task t_k. Every round in $\hat{I}_\tau(x)$ belongs to a task $t_{k'}$ with $k' \leq k$ and carries

at least one record whose touched set meets $D(x)$. Each such record is counted by $m_k(x)$, so $|I_\tau(x)| \le |\hat{I}_\tau(x)| \le m_k(x)$. Taking the supremum over x gives $h(\tau) \le m_k$. □

The count m_k is a property of how a method reads data. It does not depend on noise. The next result turns it into a privacy cost for any method whose releases are Gaussian mechanisms. It uses Rényi differential privacy (Mironov, 2017). A mechanism is $(\alpha, \epsilon_\alpha)$-RDP when the Rényi divergence of order α between its output laws on neighbouring datasets is at most ϵ_α.

Proposition 11 (The ledger accountant is sound)**.** *Fix a unit u, a sensitivity $\Delta_u > 0$, and a noise multiplier $z > 0$. Suppose every record ρ is released as $g_\rho + \xi_\rho$. Here g_ρ is a function of the prefix and of the data with $\|g_\rho(\cdot, \mathcal{D}) - g_\rho(\cdot, \mathcal{D}')\|_2 \le \Delta_u$ for every prefix and every $\mathcal{D} \sim_u \mathcal{D}'$, and $\xi_\rho \sim \mathcal{N}(0, z^2\Delta_u^2\mathbb{I})$ is fresh noise. Suppose also that touched sets obey the declaration rule. Then for every $\delta \in (0,1)$ the transcript up to task t_k is $(\epsilon_u(t_k), \delta)$-DP under $\sim_u$, with $\epsilon_u(t_k)$ given by Equation equation 5.*

Proof. Fix $\mathcal{D} \sim_u \mathcal{D}'$ that differ in the instance x, and fix an order $\alpha > 1$.

Step 1 (records that do not read x). Order the records by round and, within a round, by client. If $\omega(\rho) \cap D(x) = \emptyset$, the argument of Lemma 10 applied to the single record ρ shows that its conditional law given all earlier records is the same under $\mathcal{D}$ and $\mathcal{D}'$. The Rényi divergence between equal laws is 0 at every order.

Step 2 (records that read x). If $\omega(\rho) \cap D(x) \ne \emptyset$, the two conditional laws are Gaussians with covariance $z^2\Delta_u^2\mathbb{I}$ whose means differ by at most Δ_u. The Rényi divergence of order α between $\mathcal{N}(\mu_1, s^2\mathbb{I})$ and $\mathcal{N}(\mu_2, s^2\mathbb{I})$ is $\alpha\|\mu_1 - \mu_2\|_2^2/(2s^2)$ (Mironov, 2017). With $s = z\Delta_u$ this is at most $\alpha/(2z^2)$.

Step 3 (composition). Rényi divergence is additive under adaptive composition (Mironov, 2017). By Steps 1 and 2, the sequence of records up to task t_k is $(\alpha, m_k(x)\alpha/(2z^2))$-RDP for the pair that differs in x. Since $m_k(x) \le m_k$, it is $(\alpha, m_k\alpha/(2z^2))$-RDP for every such pair.

Step 4 (conversion). An $(\alpha, \epsilon_\alpha)$-RDP mechanism is $(\epsilon_\alpha + \log(1/\delta)/(\alpha - 1), \delta)$-DP (Mironov, 2017). Step 3 holds for every $\alpha > 1$, so the minimum over α in Equation equation 5 is a valid ϵ.

Step 5 (closed form). Write $a = m_k/(2z^2)$, $b = \log(1/\delta)$ and $v = \alpha - 1 > 0$. The objective is $a + av + b/v$. It is convex in v and is minimised at $v = \sqrt{b/a}$, where it equals $a + 2\sqrt{ab}$. Substituting a and b gives the right side of Equation equation 5.

Step 6 (transcript). Each broadcast A^{r_i} is a function of the records of round r_i and of the prefix. The transcript is therefore post-processing of the record sequence and inherits its guarantee. □

For the non-private methods we audit, the accountant answers a counterfactual question. It reports what the same pattern of data access would cost if every release were privatised at noise multiplier z. This is the right comparison, because the count m_k is fixed by the retention operator and no choice of noise changes it. The regime label on line 19 of Algorithm 2 is a finite-horizon diagnostic. A constant count over finitely many tasks does not prove that the count stays bounded forever. For our operator, Theorem 6 gives the structural proof.

A.5 The audit envelope

A weak attack does not show that little leaks. For a mechanism with a guarantee, the next result gives the ceiling that no attack can exceed. It is the hypothesis-testing form of differential privacy (**?**), applied to the pooled test of Algorithm 3.

Proposition 12 (Audit envelope)**.** *Suppose the transcript is (ϵ, δ)-DP under U_1. Fix any decision rule Γ that maps a view to $\{0,1\}$ and that does not depend on the evaluation shadows. Let* TPR *and* FPR *be the probabilities that $\Gamma = 1$ on an evaluation shadow that contains, respectively does not contain, a target drawn uniformly from $\mathcal{Z}$. Then* $\mathrm{TPR} \le e^\epsilon \mathrm{FPR} + \delta$.

Proof. Step 1. Fix a target z and the membership bits b_{-z} of all other records. The training sets with $b_z = 1$ and $b_z = 0$ differ in exactly the record z, so they are neighbours under U_1. Apply Equation equation 3 to the event $\{\Gamma = 1\}$: $\Pr[\Gamma = 1 \mid b_z = 1, b_{-z}] \le e^\epsilon \Pr[\Gamma = 1 \mid b_z = 0, b_{-z}] + \delta$.

Step 2. The bits are drawn independently, so b_{-z} has the same law whether $b_z = 1$ or $b_z = 0$. Take the expectation of both sides of Step 1 over b_{-z}. The inequality is preserved, and it becomes $\mathrm{TPR}_z \le e^{\epsilon}\mathrm{FPR}_z + \delta$, where TPR_z and FPR_z are the rates for the target z.

Step 3. Average both sides over z drawn uniformly from $\mathcal{Z}$. The right side is affine in FPR_z, so the average of the right sides equals $e^{\epsilon}\mathrm{FPR} + \delta$. This gives the claim.

Step 4. The rule Γ may depend on the calibration shadows. Those are independent of the evaluation shadows, so Steps 1 to 3 hold conditionally on them, and the claim follows by taking the expectation over the calibration shadows. □

A thresholded likelihood ratio from Algorithm 3 is such a rule for every fixed threshold. The empirical curve of true against false positive rates must therefore lie below the envelope $e^{\epsilon}\,\mathrm{FPR} + \delta$, up to sampling error. We use this in two ways. For a mechanism with a guarantee, a curve that crosses the envelope beyond its confidence interval exposes an implementation error. For any mechanism, the ratio $\log\big((\mathrm{TPR} - \delta)/\mathrm{FPR}\big)$, computed with one-sided Clopper-Pearson bounds, gives an empirical lower bound on ϵ at each threshold. Targets within one shadow are not independent, and the maximum over thresholds is not a simultaneous bound. We therefore report this lower bound as a diagnostic, not as a certificate.

A.6 Proofs for our operator

Lemma 13 (Sensitivity)**.** *Define the sensitivity of the aggregate statistic under unit u as*

$$\Delta_u = \sup_{\mathcal{D}\sim_u\mathcal{D}'} \Big\| \textstyle\sum_{c_j} \bar{S}^{t_k}_{c_j}(\mathcal{D}) - \sum_{c_j} \bar{S}^{t_k}_{c_j}(\mathcal{D}') \Big\|_2,$$

where $\bar{S}^{t_k}_{c_j}$ is the stacked statistic after any client clipping. Then $\Delta_{U_1} \le \sqrt{B^4 + B^2}$ and $\Delta_{U_2} \le C$.

Proof. Step 1 (U_1, without client clipping). Under the add-or-remove convention of Section 3.2, neighbours differ by one example (x, y) of one client. The statistic changes by the stacked pair $(\bar{x}\bar{x}^\top, \bar{x}y^\top)$. Its squared norm is $\|\bar{x}\bar{x}^\top\|_F^2 + \|\bar{x}y^\top\|_F^2 = \|\bar{x}\|_2^4 + \|\bar{x}\|_2^2\|y\|_2^2$. Since $\|y\|_2 = 1$ and $\|\bar{x}\|_2 \le B$ by Equation equation 7, the norm is at most $\sqrt{B^4 + B^2}$. The statistics of all other clients are unchanged.

Step 2 (U_1, with client clipping). Client clipping is the Euclidean projection onto the ball of radius C. Projection onto a closed convex set is non-expansive, so the clipped contribution changes by no more than the unclipped one. The bound of Step 1 still holds.

Step 3 (U_2). Neighbours differ in the entire task-t_k contribution of one client c_j. The aggregate changes by $\bar{S}^{t_k}_{c_j}$, whose norm is at most C after clipping. □

Theorem 6 (Retaining the release is free)**.** *Let $u \in \{U_1, U_2\}$ and let σ be calibrated to Δ_u so that one Gaussian release is (ϵ_0, δ_0)-DP. Then the transcript $(A^{r_1}, A^{r_2}, \dots)$ of Algorithm 1 is (ϵ_0, δ_0)-DP under u over the whole unbounded stream. The classifiers $(\tilde{W}^{t_1}, \tilde{W}^{t_2}, \dots)$ satisfy the same guarantee, for every decay $\gamma \in [0, 1]$ and every ridge parameter $\lambda > 0$.*

Proof. Step 1 (horizon). Fix a unit instance x. Under U_1 it is one example of some $D^{t_k}_{c_j}$. Under U_2 it is $D^{t_k}_{c_j}$ itself. In both cases x belongs to the data of exactly one task t_k. By Equation equation 8, the release $A^{r_{k'}}$ depends only on the data of task $t_{k'}$, on the fixed public projection P, and on fresh noise $\xi^{r_{k'}}$. It depends neither on the prefix nor on the retained state. So for $k' \ne k$ the law of $A^{r_{k'}}$ is the same under $\mathcal{D}$ and $\mathcal{D}'$. By Definition 1, $I_\tau(x) \subseteq \{r_k\}$ for every τ. The operator is contractive with $h = 1$.

Step 2 (one release). By Lemma 13, the aggregate statistic has ℓ_2 sensitivity at most Δ_u under $\sim_u$. The analytic Gaussian mechanism with $\sigma = \sigma_{\mathrm{AG}}(\epsilon_0, \delta_0, \Delta_u)$ is (ϵ_0, δ_0)-DP (Balle & Wang, 2018). So the release at task t_k satisfies condition (A1) of Theorem 2.

Step 3 (composition). Conditions (A1) and (A2) of Theorem 2 hold with $h = 1$. The theorem gives $(1 \cdot \epsilon_0, 1 \cdot \delta_0)$ for the unbounded transcript.

Step 4 (retention is post-processing). By Equation equation 8, $\tilde{R}^{t_k}$, $\tilde{Q}^{t_k}$, and $\tilde{W}^{t_k}$ are deterministic functions of $(A^{r_1}, \dots, A^{r_k})$ and of the constants γ and λ. Differential privacy is closed under post-processing, so the classifiers inherit the guarantee of Step 3. Neither γ nor λ enters any release, so the guarantee does not depend on them. □

Corollary 7 (Other units)**.** *Calibrate σ to Δ_{U_2}. Under U_3 with a window of W tasks, the transcript of Algorithm 1 is $(W\epsilon_0, W\delta_0)$-DP. Under U_4, a client that contributes to infinitely many tasks has no finite lifelong guarantee, for this or any mechanism covered by Proposition 3.*

Proof. Step 1 (U_3). A unit instance is one client's data over at most W tasks. By Step 1 of the proof of Theorem 6, each of those tasks is read by exactly one release. So $|I_\tau(x)| \le W$ for every τ.

Step 2. Each influenced release changes by at most one client-task contribution, whose norm is at most C by Lemma 13. Each is therefore (ϵ_0, δ_0)-DP. Theorem 2 with $h = W$ gives $(W\epsilon_0, W\delta_0)$.

Step 3 (U_4). A unit instance is one client's data over the whole stream. If the client participates in tasks $t_{k_1}, t_{k_2}, \dots$ with $k_1 < k_2 < \cdots$, the release of every such task reads its data. So $|I_\tau(x)| \to \infty$ and the operator is accumulating under U_4. The releases are additive Gaussian and do not depend on the prefix. Under the non-degeneracy of Proposition 3, it gives $\epsilon(\tau) \to \infty$. □

The calibration in the corollary matters. If σ is calibrated to Δ_{U_1} only, Theorem 6 certifies U_1 and nothing more. A guarantee under U_2, U_3 or U_4 then requires group privacy over every example of a client, and a count of releases does not supply it. Corollary 7 is also a statement about the problem, not about our operator. No additive Gaussian mechanism can promise lifelong privacy to a client that keeps contributing forever.

A.7 Bounded error for a decay below one

Privacy alone does not make an operator useful. The next result shows that the noise in the retained state stays bounded when $\gamma < 1$, so accuracy does not decay as the stream grows. We compare against the non-private operator with the same projection, decay and clipping, whose state (R^{t_k}, Q^{t_k}) and classifier $W^{t_k} = (R^{t_k} + \lambda \mathbb{I})^{-1} Q^{t_k}$ follow Equation equation 8 with $\xi^{r_k} = 0$. This comparison isolates the cost of privacy from the choice of γ.

Proposition 14 (Bounded error for $\gamma < 1$)**.** *Suppose $\|H^{t_k}\|_F \le \eta$ for every task t_k, where H^{t_k} is the non-private aggregate of the second statistic of Equation equation 7 after clipping. Then for every $\gamma \in [0, 1)$ and every k,*

$$\mathbb{E}\left\|\tilde{W}^{t_k} - W^{t_k}\right\|_F \;\le\; \frac{\sigma}{\sqrt{1-\gamma^2}} \left(\frac{\sqrt{qJ}}{\lambda} + \frac{q\,\eta}{\lambda^2(1-\gamma)} \right), \tag{10}$$

where J is the number of classes. The right side does not depend on k.

Proof. Step 1 (noise in the retained state). Write $(\Xi_G^{r_{k'}}, \Xi_H^{r_{k'}})$ for the noise $\xi^{r_{k'}}$ reshaped into the two blocks. The recursion in Equation equation 8 is linear, so the noise in the state is $E_R^{t_k} = \tilde{R}^{t_k} - R^{t_k} = \sum_{k'=1}^{k} \gamma^{k-k'} \, \frac{1}{2}(\Xi_G^{r_{k'}} + \Xi_G^{r_{k'}\top})$ and $E_Q^{t_k} = \tilde{Q}^{t_k} - Q^{t_k} = \sum_{k'=1}^{k} \gamma^{k-k'} \, \Xi_H^{r_{k'}}$.

Step 2 (second moments). The noise terms are independent across k' and have mean zero, so cross terms vanish in expectation. Every entry of $\Xi_H^{r_{k'}}$ has variance σ^2, so

$$\mathbb{E}\|E_Q^{t_k}\|_F^2 = qJ\sigma^2 \sum_{k'=1}^{k} \gamma^{2(k-k')} \le \frac{qJ\sigma^2}{1-\gamma^2}.$$

By the triangle inequality, $\|\frac{1}{2}(\Xi + \Xi^\top)\|_F \le \|\Xi\|_F$. The same argument therefore gives $\mathbb{E}\|E_R^{t_k}\|_F^2 \le q^2\sigma^2/(1-\gamma^2)$.

Step 3 (projection). The matrix R^{t_k} is a non-negative combination of positive semidefinite matrices, so it is positive semidefinite and $\Pi_+(R^{t_k}) = R^{t_k}$. The projection Π_+ onto a closed convex set is non-expansive in the Frobenius norm. Hence $\|\Pi_+(\tilde{R}^{t_k}) - R^{t_k}\|_F \le \|E_R^{t_k}\|_F$.

Step 4 (perturbation of the solution). Let $\tilde{\mathbf{A}} = \Pi_+(\tilde{R}^{t_k}) + \lambda\mathbb{I}$ and $\mathbf{A} = R^{t_k} + \lambda\mathbb{I}$. Both satisfy $\succeq \lambda\mathbb{I}$, so their inverses have spectral norm at most $1/\lambda$. Using $\tilde{\mathbf{A}}^{-1} - \mathbf{A}^{-1} = \tilde{\mathbf{A}}^{-1}(\mathbf{A} - \tilde{\mathbf{A}})\mathbf{A}^{-1}$,

$$\tilde{W}^{t_k} - W^{t_k} = \tilde{\mathbf{A}}^{-1}\big(\tilde{Q}^{t_k} - Q^{t_k}\big) + \tilde{\mathbf{A}}^{-1}\big(\mathbf{A} - \tilde{\mathbf{A}}\big)\mathbf{A}^{-1}Q^{t_k}.$$

Applying $\|XY\|_F \le \|X\|_2\|Y\|_F$ and $\|X\|_2 \le \|X\|_F$ together with Step 3,

$$\|\tilde{W}^{t_k} - W^{t_k}\|_F \le \frac{1}{\lambda}\|E_Q^{t_k}\|_F + \frac{1}{\lambda^2}\|E_R^{t_k}\|_F\,\|Q^{t_k}\|_F.$$

Step 5 (the signal term). By the recursion and the triangle inequality,

$$\|Q^{t_k}\|_F \le \sum_{k'=1}^{k} \gamma^{k-k'}\|H^{t_{k'}}\|_F \le \frac{\eta}{1-\gamma}.$$

Step 6 (expectation). Q^{t_k} is not random. Take expectations in Step 4, bound $\|Q^{t_k}\|_F$ by Step 5, and apply $\mathbb{E}\|E\|_F \le (\mathbb{E}\|E\|_F^2)^{1/2}$ to both noise terms using Step 2. This gives Equation equation 10. No step introduces a dependence on k. □

Setting $\gamma = 1$ in Steps 2 and 5 shows the alternative. The noise variance then grows linearly in k, and the bound on the signal term grows as well. The bound on the error of a perfectly retaining noisy operator therefore grows without limit. The decay γ therefore plays two roles. It trades retention against plasticity exactly as in the non-private method, and it keeps the accumulated noise bounded. By Theorem 6, neither role touches the privacy guarantee.

B THE FRAMEWORK AND THE MEASUREMENT ALGORITHMS

This appendix describes the full system and states the two measurements of Section 4.1 as algorithms.

System overview. Figure 3 shows the framework. It has four stages, and information moves through them in one direction. In the federation, a participating client c_j computes an artifact $a_{c_j}^{r_i}$ from its current data $D_{c_j}^{t_k}$ and from any local state it keeps, and declares the touched set ω of that release. The server combines the uploads into the broadcast A^{r_i} and applies the retention operator Ret, which decides what is carried from one task to the next. Methods differ mainly in this operator, and Theorem 2 and Proposition 3 show that privacy is decided here as well. The release channel $\mathcal{M}$ holds the transcript $\mathcal{A}$, which a passive adversary with view $\mathcal{V}_{\mathrm{full}}$ observes, and the ledger $\mathcal{R}$, which stores every release with its family and touched set. In the analysis stage, the accountant counts how many releases read the data of each unit instance and converts the count into $\epsilon_u(t_k)$ for every task t_k. The audit measures accuracy and leakage about a fixed task as a function of elapsed tasks. Every analysis reads only the release channel. This rule lets one audit cover methods whose internals share nothing, such as replay buffers and closed-form statistics.

The declaration rule. The touched set carries the whole privacy argument. A datum held in a local replay buffer belongs in $\omega(\rho)$ whenever the computation of ρ trains on that buffer. A broadcast model used as a distillation teacher, a released statistic, or any quantity computed only from $\mathcal{A}$ adds nothing to $\omega(\rho)$, because its value is already public. If a method cannot declare $\omega(\rho)$, the release is treated as reading every datum the client holds. Under-reporting ω would turn a false privacy claim into a certified one, and we prefer a vacuous certificate to a wrong one.

Units of privacy. Table 5 defines the five units of Section 3.2. We use the add-or-remove convention. Under the replacement convention every sensitivity in this paper doubles, and every statement holds with that change.

B.1 INSTRUMENTED TRAINING AND LIFELONG ACCOUNTING

Algorithm 2 trains a federated continual learner for K tasks. It produces the transcript, the ledger, and the accuracy matrix in one pass. It then counts, for every instance of a unit, the releases that read its data, and converts the counts into a privacy cost.

Table 5: Units of privacy. A guarantee under a unit protects one instance of it.

Unit	Instance x	Data $D(x)$	Protects
U_1	one example $z \in D_{c_j}^{t_k}$	$\{z\}$	a single record
U_2	one client in one task	$D_{c_j}^{t_k}$	one task of participation
U_3	one client in a window of W tasks	$\bigcup_{k'=k}^{k+W-1} D_{c_j}^{t_{k'}}$	a bounded tenure
U_4	one client over the whole stream	$\bigcup_{k \geq 1} D_{c_j}^{t_k}$	lifelong participation
U_5	one person	all records of that person	a person

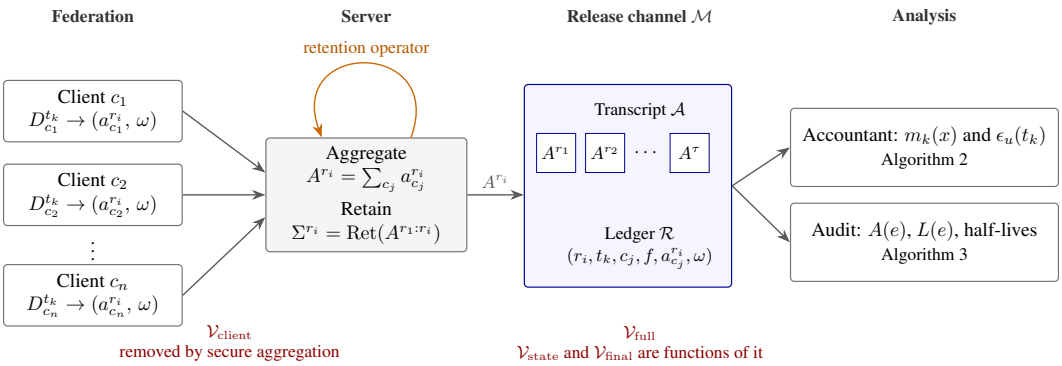


Figure 3: System framework. Clients release artifacts $a_{c_j}^{r_i}$ and declare the touched set ω of each release. The server aggregates the uploads into the broadcast A^{r_i} and carries state Σ^{r_i} across tasks through a retention operator. The release channel $\mathcal{M}$ records the transcript $\mathcal{A}$ and the ledger $\mathcal{R}$. Both analyses read only the release channel, never the internal state of a method. Secure aggregation removes the per-client view and leaves the transcript, the retained state, and the final model intact. Our operator (Algorithm 1) changes one component: its retention operator reads past releases and never past data.

Line 1 initialises the round index i, the ledger, and the transcript. Lines 2 to 11 are the ordinary training loop. Line 4 iterates over the clients that take part in the round. Line 5 is the only change to standard practice. The method must return its artifact, the family of that artifact, and the touched set. A method that releases several families in one round returns one record per family. Line 6 appends the records to the ledger. Line 8 aggregates the uploads into the broadcast and appends it to the transcript. Line 10 evaluates the current model on the held-out test data of every task seen so far. The prediction is restricted to classes that have already appeared, as is standard for class-incremental evaluation. Line 10 fills row k of the accuracy matrix M, where $M_{k,k'}$ is the accuracy on task $t_{k'}$ after task t_k is finished.

Lines 12 to 14 enforce the declaration rule of Section 3.2. A method that declares nothing certifies nothing. Lines 15 to 18 are the accountant. Line 16 counts, for every instance x of the unit, the releases up to task t_k whose touched set meets the data $D(x)$ of that instance. Here $k(\rho)$ is the index of the task of record ρ. Line 17 takes the largest count and converts it into a privacy cost at noise multiplier z. Line 19 labels the run CONTRACTIVE when the count stops growing over the second half of the stream, and ACCUMULATING otherwise.

Lemma 10 in Appendix A.4 shows that the count on line 16 bounds the influence horizon, and Proposition 11 shows that line 17 is a valid privacy cost for Gaussian releases.

B.2 THE TRANSCRIPT AUDIT

The audit measures how much an adversary can still learn about the data of task t_k after the federation has reached task t_{k+e}. It adapts the likelihood-ratio membership test of Carlini et al. (2022a) in two ways. First, the attack statistic is computed from a view of the channel rather than from a single model. Second, the target task and the observation time vary independently. The second change produces a leakage curve over elapsed tasks.

Algorithm 2 Instrumented federated continual learning with lifelong accounting

Require: method Ω; clients $\mathcal{C}$; task stream $\{D_{c_j}^{t_k}\}_{k=1}^{K}$; test sets $\{D_{\text{test}}^{t_k}\}_{k=1}^{K}$; unit u; noise multiplier z; target δ

Ensure: transcript $\mathcal{A}$; ledger $\mathcal{R}$; accuracy matrix M; counts m_k and costs $\epsilon_u(t_k)$ for $k = 1, \ldots, K$; regime

1: $i \leftarrow 1$; $\mathcal{R} \leftarrow \emptyset$; $\mathcal{A} \leftarrow ()$
2: **for** $k = 1, \ldots, K$ **do**
3: **for** each round $r_i \in R_k$ **do**
4: **for** $c_j \in \mathcal{C}_{r_i}$ **do**
5: $(a_{c_j}^{r_i},\ f,\ \omega) \leftarrow \Omega.\textsc{LocalUpdate}(c_j, t_k, r_i, D_{c_j}^{t_k})$
6: $\mathcal{R} \leftarrow \mathcal{R} \cup \{(r_i, t_k, c_j, f, a_{c_j}^{r_i}, \omega)\}$
7: **end for**
8: $A^{r_i} \leftarrow \Omega.\textsc{Aggregate}(\{a_{c_j}^{r_i}\}_{c_j \in \mathcal{C}_{r_i}})$; append A^{r_i} to $\mathcal{A}$; $i \leftarrow i + 1$
9: **end for**
10: $M_{k,k'} \leftarrow$ accuracy of Ω on $D_{\text{test}}^{t_{k'}}$, predicting among classes seen so far, for $k' = 1, \ldots, k$
11: **end for**
12: **if** some $\rho \in \mathcal{R}$ has no declared $\omega(\rho)$ **then**
13: **return** UNVERIFIABLE
14: **end if**
15: **for** $k = 1, \ldots, K$ **do**
16: $m_k(x) \leftarrow \big|\{\rho \in \mathcal{R} : k(\rho) \le k,\ \omega(\rho) \cap D(x) \neq \emptyset\}\big|$ for every instance x of unit u
17: $m_k \leftarrow \max_x m_k(x)$; $\epsilon_u(t_k) \leftarrow \min_{\alpha>1}\big[m_k\,\alpha/(2z^2) + \log(1/\delta)/(\alpha - 1)\big]$
18: **end for**
19: **return** CONTRACTIVE if m_k is constant for $k \ge \lceil K/2 \rceil$, and ACCUMULATING otherwise

Scores. A score maps a target record $z = (x_z, y_z)$ and the content of a view at round r_i to a real number. We use one score per artifact family. For model deltas (F_1), the view contains the broadcast linear head W^{r_i}, and the score is the logit margin $s(z; W^{r_i}) = (W^{r_i\top} x_z)_{y_z} - \max_{y \neq y_z} (W^{r_i\top} x_z)_y$. For prototypes ($F_2$), the view contains a prototype μ_{y_z} of the target's class, and the score is $s(z; \mu) = -\|x_z - \mu_{y_z}\|_2$. For second-moment statistics (F_5), the view contains a matrix R, and the score is the leverage $s(z; R) = x_z^\top R^{-1} x_z$. Our operator uses the projected feature $\bar{x}_z$ of Section 4.2 in place of x_z. The prototype and the matrix come from the client's own release under $\mathcal{V}_{\text{client}}$, from the aggregate A^{r_i} under $\mathcal{V}_{\text{full}}$, and from the retained state Σ^{r_i} under $\mathcal{V}_{\text{state}}$. The likelihood-ratio test below learns the direction of each score from shadow runs, so no score needs to increase with membership.

Rounds. Let $t_{k(z)}$ be the task of target z. For elapsed tasks $e \ge 0$, the transcript adversary reads every round from the start of task $t_{k(z)}$ to the end of task $t_{k(z)+e}$. The checkpoint adversary reads only the last of these rounds. We write $\mathcal{T}_e(z)$ for the set of rounds that the chosen adversary reads.

Lines 1 to 5 train N shadow federations. Line 2 draws a fresh membership bit for every record in the training pool, not only for the targets. Each shadow is therefore an independent draw of the whole federation. Line 3 runs the full procedure of Algorithm 2, so shadows produce the same kind of ledger as a real run. Line 4 records the score of every target at every round. One pass over the ledger serves every elapsed time at once, which makes the two-index sweep affordable. The shadows are independent and run in parallel.

Line 6 splits the shadows. The calibration shadows play the role of the attacker's reference runs. The evaluation shadows play the role of the target federation. They are exchangeable with the calibration shadows, and their membership bits are known, so every evaluation shadow is a fresh test case with ground truth. Line 7 fits two Gaussians for each target and each round. One describes the scores of calibration shadows that contained the target. The other describes the scores of those that did not. Lines 8 to 11 score each target in each evaluation shadow by its log-likelihood ratio. The ratio is summed over the rounds that the adversary reads. The lines then read off the true positive rate at a fixed false positive rate β. We fix β at a low value, following Carlini et al. (2022a), because an attack that works only at high false positive rates tells a data owner little. Line 12 measures accuracy on the unmodified federation. Lines 13 and 14 normalise both curves and compute their half-lives.

Algorithm 3 Cross-task transcript audit

Require: method Ω; stream of K tasks; target tasks $\mathcal{K} \subseteq T$ and horizon E with $k+E \leq K$ for every $t_k \in \mathcal{K}$; targets $\mathcal{Z}$ from tasks in $\mathcal{K}$; score s_f; view $\mathcal{V}$; adversary TRANSCRIPT or CHECKPOINT; shadows N; calibration fraction κ; level β

Ensure: normalised curves $A(e)$ and $L(e)$ for $e = 0, \ldots, E$; half-lives h_A, h_L; ratio ϱ

1: **for** $s = 1, \ldots, N$ **do**
2: draw $b_z^s \sim \text{Bernoulli}(1/2)$ independently for every record z of the training pool
3: run Algorithm 2 on the records with $b_z^s = 1$; keep the ledger $\mathcal{R}^s$
4: $\phi_{r_i}^s(z) \leftarrow s_f\big(z; \mathcal{V}_{r_i}(\mathcal{R}^s)\big)$ for every $z \in \mathcal{Z}$ and every round r_i from the start of task $t_{k(z)}$
5: **end for**
6: split the shadows into a calibration set $\mathcal{S}_\text{c}$ of size $\lceil \kappa N \rceil$ and an evaluation set $\mathcal{S}_\text{e}$
7: for every z and r_i, fit $\mathcal{N}(\mu_\text{in}, \sigma_\text{in}^2)$ to $\{\phi_{r_i}^s(z) : s \in \mathcal{S}_\text{c},\ b_z^s = 1\}$ and $\mathcal{N}(\mu_\text{out}, \sigma_\text{out}^2)$ to $\{\phi_{r_i}^s(z) : s \in \mathcal{S}_\text{c},\ b_z^s = 0\}$
8: **for** $e = 0, \ldots, E$ **do**
9: $\Lambda_e^s(z) \leftarrow \sum_{r_i \in \mathcal{T}_e(z)} \log \dfrac{\mathcal{N}(\phi_{r_i}^s(z);\, \mu_\text{in}, \sigma_\text{in}^2)}{\mathcal{N}(\phi_{r_i}^s(z);\, \mu_\text{out}, \sigma_\text{out}^2)}$ for every $s \in \mathcal{S}_\text{e}$ and $z \in \mathcal{Z}$
10: $\text{TPR}_\beta(e) \leftarrow$ true positive rate of $\{\Lambda_e^s(z)\}$ against $\{b_z^s\}$ at false positive rate β, pooled over $s \in \mathcal{S}_\text{e}$ and $z \in \mathcal{Z}$
11: **end for**
12: run Algorithm 2 on the full pool over several seeds and average the accuracy matrices into M
13: compute $A(e)$ and $L(e)$ for $e = 0, \ldots, E$ by Equation equation 6
14: compute h_A, h_L and ϱ by Definition 5
15: **return** A, L, h_A, h_L, ϱ

Two design choices keep the curves honest. First, the target tasks $\mathcal{K}$ are fixed in advance and satisfy $k + E \leq K$ for every $t_k \in \mathcal{K}$. The same targets are therefore tested at every elapsed time, and a change in the curve cannot come from a change in the population being tested. Second, both curves are measured against their chance level and normalised at the end of the target task, as in Equation equation 6. There $a_0(t_k) = 1/|\mathcal{Y}_k|$, where $\mathcal{Y}_k$ is the set of classes seen after task t_k, and β is also the true positive rate of a random scorer at false positive rate β. A value of 0 means the signal is at chance. We attach 95% intervals to both curves by a hierarchical bootstrap. It resamples seeds first, and then targets for L or target tasks for A.

We report a half-life only when the denominator of ψ in Equation equation 6 is positive with confidence. The decoupling ratio is a lower bound when only h_L is censored, and it is undefined when h_A is censored. When a method keeps its accuracy, h_A is censored, and the pair $(A(E), L(E))$ carries the comparison. The definition assumes no decay law and never extrapolates beyond the observed horizon.

B.3 OUR OPERATOR IN DETAIL

In Algorithm 1, line 1 calibrates the noise with the analytic Gaussian mechanism (Balle & Wang, 2018), which is valid for every ϵ_0, and initialises the retained state at zero. Line 4 projects each feature onto the public basis and clips it, so that no single example can move the statistics by more than a known amount. Under U_2 it also clips the whole contribution of the client. Line 5 declares the touched set, which contains the data of task t_k and nothing else. Line 7 turns the statistics into the single release that depends on task-t_k data. Line 8 symmetrises the noisy second-moment matrix. Line 9 applies the retention operator with decay γ and solves for the classifier after projecting onto the positive semidefinite cone, which keeps the linear system well posed. Line 10 discards the data and the unreleased statistics. The projection P and the settings q, λ and C are chosen on the public reference split, which shares no record with any client, so these choices cost no privacy.

The noise can be added by a trusted aggregator. When the server itself is the adversary, each participating client adds an independent share $\mathcal{N}(0, \sigma^2 \mathbb{I}/|\mathcal{C}_{r_k}|)$, and the shares are summed under secure aggregation (Bonawitz et al., 2017). The sum has the law of the release in Equation equation 8, so every guarantee covers the server as well.

C IMPLEMENTATION DETAILS

Data. Table 6 gives the sizes of the three benchmarks. Each dataset is split into a training set, a test set, and a public reference set. The reference set shares no image with any client. It is used for three purposes only: to fit the projection of our operator, to select its settings, and as the public data of the reconstruction test in Appendix E. Features are the class token of the ViT-B/16 taken before its final normalisation layer. The pretraining data of the backbone overlap the classes of ImageNet-R, which can make that benchmark easier. Predictions are restricted to the classes seen so far.

Table 6: Datasets. Each dataset is split into ten tasks with disjoint classes. The last column is the average number of training examples per class.

Dataset	Classes	Classes per task	Train	Test	Public reference	Per class
CIFAR-100	100	10	40,000	10,000	10,000	400
CUB-200-2011	200	20	4,994	5,794	1,000	25
ImageNet-R	200	20	23,918	2,994	3,088	120

Table 7: Methods. The families refer to Section 3.2: model updates (F_1), prototypes (F_2), second-moment statistics (F_5), generative statistics (F_6), and per-class counts (F_7).

ID	Method	What it retains across tasks	Releases
M0	FedAvg	Nothing. It fine-tunes the shared classifier on each new task.	F_1
M1	GLFC-style (Dong et al., 2022a)	A private buffer of 10 examples per class on each client, plus distillation from the previous model on old classes.	F_1, F_7
M2	Gaussian replay (Zhang et al., 2023)	Per-class feature means and variances, released once and sampled for replay (20 samples per class).	F_1, F_6
M3	FOT (Bakman et al., 2024)	A feature subspace of old tasks. Later updates are projected away from it.	F_1, F_5
M4	Prototypes (Guo et al., 2024)	Per-class feature means and counts. The server keeps their count-weighted average.	F_2, F_7
M5	Exemplar replay (Khademi Nori et al., 2025)	A private buffer of 10 examples per class on each client.	F_1
M8	Analytic (Tang et al., 2026)	Running sums of second-moment statistics, solved in closed form.	F_5
Ours	Algorithm 1	Noisy second-moment statistics, carried forward with decay γ.	F_5

Methods. Table 7 lists the methods, what each retains, and the families it releases. By default, the three replay methods give the loss on current data and the loss on replayed data equal weight.

Uncertainty and accounting. For leakage we resample seeds and then targets. For accuracy we resample seeds and then target tasks. For the half-life ratio of M0 we resample seeds, tasks and targets jointly, so that both half-lives come from the same draw. The accountant runs on the ledger of one CIFAR-100 run of each method. We extend each count beyond the ten observed tasks in the simplest way. A count that is constant over the second half of the stream stays constant. A count that grows is extended linearly.

Shared protocol. All methods share the protocol of Section 5. They use frozen ViT-B/16 features, ten tasks, and ten clients with a Dirichlet split of parameter 0.5. Each task has one round of federated averaging, in which every client runs 30 epochs of full-batch gradient descent. The learning rate is 0.5 unless Table 8 states otherwise.

Fixed method settings. M1 keeps up to 10 examples per class on each client, with distillation weight 1 and temperature 2. M2 replays 20 synthetic samples per class. M3 uses a subspace rank of 8

and a projection strength of 1. M4 averages prototypes weighted by their counts, with no momentum. M5 keeps up to 10 examples per class on each client. M8 uses the ridge parameter $\lambda = 1$. The dose study of Section **??** varies only the buffer of M5 and the number of synthetic samples of M2.

A check on the replay methods. A replay method can reach high accuracy on old tasks simply by failing to learn new ones. We therefore check each method against FedAvg on the same stream, averaged over three seeds. A method passes when three conditions hold. Its backward transfer is at most 0.02. Its accuracy on the last task is at most 0.15 below that of FedAvg. Its final average accuracy is at least that of FedAvg. The first condition excludes methods whose accuracy on old tasks keeps rising after the task ends. The second requires that the method still learns new tasks. When a replay method failed with its default settings, we searched the replay weight over $\{0.5, 1, 2\}$ and the learning rate over $\{0.1, 0.5\}$. The search used a stratified 10% validation split of the training data and never the test data. We kept a setting that passed on the validation split when one existed, and otherwise the setting with the highest validation accuracy. Table 8 reports the outcome.

Table 8: The check on the replay methods, averaged over three seeds. A dagger marks a setting chosen on the validation split because the default failed. BWT is backward transfer. "Newest" is the accuracy on the last task at the end of the stream. "Final" is the final average accuracy over all tasks.

Dataset	Method	Learning rate	Replay weight	BWT	Newest	Final	Passes
CIFAR-100	M0 FedAvg	0.5	none	−0.670	0.951	0.333	reference
	M1 GLFC-style	0.5†	2†	0.003	0.798	0.843	no
	M2 Gaussian replay	0.5	1	−0.033	0.838	0.836	yes
	M3 FOT	0.5	none	−0.174	0.920	0.748	yes
	M5 Exemplar replay	0.5	1	−0.043	0.853	0.841	yes
CUB-200-2011	M0 FedAvg	0.5	none	−0.525	0.836	0.386	reference
	M1 GLFC-style	0.5†	2†	0.360	0.129	0.697	no
	M2 Gaussian replay	0.1†	2†	0.481	0.171	0.787	no
	M3 FOT	0.5	none	−0.008	0.695	0.671	yes
	M5 Exemplar replay	0.1†	2†	0.314	0.282	0.779	no
ImageNet-R	M0 FedAvg	0.5	none	−0.526	0.687	0.248	reference
	M1 GLFC-style	0.5†	2†	0.117	0.382	0.598	no
	M2 Gaussian replay	0.1†	0.5†	0.098	0.444	0.616	no
	M3 FOT	0.5	none	−0.269	0.657	0.452	yes
	M5 Exemplar replay	0.1†	2†	0.065	0.461	0.631	no

Only M2 and M5 on CIFAR-100 pass among the replay methods. M1 on CIFAR-100 misses the second condition narrowly, with a last-task accuracy of 0.798 against a threshold of 0.801. On CUB-200-2011 and ImageNet-R all three replay methods fail both of the first two conditions, even after the search. Their accuracy on the newest task is far below that of FedAvg, and their backward transfer is positive. Appendix **??** discusses what this means for our leakage results. M3 passes on every dataset.

Settings of our operator. The projection dimension q is chosen from 32, 64, 128, 256 and 768, where 768 means no projection. The ridge parameter λ is chosen from 0.01, 0.1, 1, 10 and 100. The client clip C is the 95th percentile of the norms of simulated client statistics on the public reference set. Table 9 lists the values selected for each dataset, unit and privacy level. On CIFAR-100 we also run U_2 with 50 and 100 clients.

D Additional results

D.1 Accuracy of the methods

Table 10 reports final average accuracy and backward transfer over five seeds. Backward transfer averages, over all tasks but the last, the change in accuracy on a task between the end of that task and the end of the stream. A negative value is forgetting.

Table 9: Settings of our operator selected on the public reference set, written as q / λ.

Dataset	Unit	$\epsilon_0 = 0.5$	1	2	4	8	∞
CIFAR-100	U_1	64 / 100	64 / 100	64 / 100	128 / 100	128 / 100	768 / 0.01
	U_2	32 / 0.01	32 / 0.01	32 / 10	32 / 10	32 / 100	768 / 0.1
CUB-200-2011	U_1	32 / 10	64 / 100	64 / 100	64 / 10	256 / 1	768 / 1
	U_2	32 / 0.01	32 / 0.01	32 / 0.01	32 / 0.01	64 / 100	768 / 1
ImageNet-R	U_1	128 / 0.1	128 / 100	128 / 100	128 / 100	128 / 100	768 / 1
	U_2	128 / 0.01	128 / 0.01	128 / 0.01	128 / 1	128 / 100	768 / 1

Table 10: Final average accuracy (Final) and backward transfer (BWT) after ten tasks, over five seeds. Brackets give 95% intervals.

	CIFAR-100		CUB-200-2011		ImageNet-R	
Method	Final	BWT	Final	BWT	Final	BWT
M0 FedAvg	0.334 [0.304, 0.363]	−0.668 [−0.702, −0.633]	0.379 [0.324, 0.434]	−0.549 [−0.620, −0.478]	0.249 [0.223, 0.275]	−0.528 [−0.565, −0.492]
M1 GLFC-style	0.843 [0.840, 0.845]	0.004 [0.001, 0.006]	0.691 [0.681, 0.702]	0.352 [0.320, 0.385]	0.597 [0.594, 0.601]	0.122 [0.108, 0.136]
M2 Gaussian replay	0.836 [0.831, 0.841]	−0.033 [−0.037, −0.029]	0.789 [0.782, 0.796]	0.486 [0.474, 0.498]	0.615 [0.610, 0.620]	0.097 [0.093, 0.101]
M3 FOT	0.748 [0.733, 0.763]	−0.176 [−0.197, −0.156]	0.673 [0.661, 0.685]	−0.024 [−0.105, 0.058]	0.457 [0.443, 0.472]	−0.259 [−0.287, −0.231]
M4 Prototypes	0.770 [0.770, 0.770]	−0.066 [−0.071, −0.062]	0.857 [0.857, 0.857]	−0.055 [−0.060, −0.049]	0.514 [0.512, 0.517]	−0.087 [−0.093, −0.081]
M5 Exemplar replay	0.842 [0.834, 0.850]	−0.040 [−0.046, −0.033]	0.780 [0.764, 0.795]	0.321 [0.291, 0.350]	0.631 [0.621, 0.641]	0.066 [0.061, 0.072]
M8 Analytic	0.883 [0.883, 0.883]	−0.045 [−0.048, −0.042]	0.871 [0.871, 0.872]	−0.048 [−0.053, −0.044]	0.635 [0.634, 0.636]	−0.089 [−0.094, −0.084]

FedAvg forgets most of each task, with backward transfer between -0.67 and -0.53. Every other method reaches a final accuracy at least 0.2 higher on every dataset. M8 reaches the highest final accuracy on all three datasets, 0.883, 0.871 and 0.635. It solves the classifier in closed form from its statistics. The replay methods M1, M2 and M5 have positive backward transfer on CUB-200-2011 and ImageNet-R, from 0.066 to 0.486. Appendix C traces this to their low accuracy on each new task when it arrives.

D.2 Membership leakage in detail

Table 11 reports three measures of the membership attack when the target task ends and six tasks later. For model updates the attack is the transcript attacker. Figure 4 shows the full curves when the task ends, and Figure 5 shows the spread across seeds.

At the stricter false positive rate of 0.1%, the replay methods show the same growth as at 1%. For M5 the TPR rises from 0.008 to 0.024 on CIFAR-100, from 0.016 to 0.117 on CUB-200-2011, and from 0.029 to 0.159 on ImageNet-R. The area under the curve moves much less. For model updates on CIFAR-100 it lies between 0.55 and 0.63 at both times, even where the TPR doubles. It is a poor summary of the risk to the most exposed records, which is why we report the TPR at a low false positive rate as the main measure.

D.3 Accuracy and leakage at the horizon

Figure 6 plots the values of Table 2 as points, together with the per-client views. Figure 7 follows the two methods that retain statistics over time on CIFAR-100.

D.4 Dose response

Table 12 gives the values behind Figure 2 of Section **??**.

Table 11: Membership attack when the target task ends ($e = 0$) and six tasks later ($e = 6$). Top: TPR at a 1% false positive rate, where a random guess scores 0.01. Middle: TPR at a 0.1% false positive rate, where a random guess scores 0.001. Bottom: area under the ROC curve, where a random guess scores 0.5. The per-client release and the aggregate of M4 and M8 do not change after they are sent, so their two columns are equal.

	CIFAR-100		CUB-200-2011		ImageNet-R	
Method and view	$e = 0$	$e = 6$	$e = 0$	$e = 6$	$e = 0$	$e = 6$
TPR at 1% false positive rate						
M0 FedAvg	0.028	0.026	0.058	0.025	0.074	0.072
M1 GLFC-style	0.027	0.055	0.038	0.120	0.060	0.181
M2 Gaussian replay	0.032	0.042	0.076	0.329	0.095	0.235
M3 FOT	0.029	0.038	0.062	0.090	0.079	0.090
M5 Exemplar replay	0.031	0.064	0.064	0.261	0.086	0.292
M4 Prototypes, per-client	0.343	0.343	0.996	0.996	0.732	0.732
M4 Prototypes, aggregate	0.058	0.058	0.832	0.832	0.240	0.240
M8 Analytic, per-client	1.000	1.000	1.000	1.000	1.000	1.000
M8 Analytic, aggregate	0.999	0.999	1.000	1.000	1.000	1.000
M8 Analytic, state	0.966	0.917	1.000	1.000	0.996	0.984
TPR at 0.1% false positive rate						
M0 FedAvg	0.007	0.005	0.016	0.004	0.022	0.014
M1 GLFC-style	0.007	0.023	0.006	0.023	0.015	0.079
M2 Gaussian replay	0.008	0.010	0.019	0.149	0.027	0.105
M3 FOT	0.005	0.008	0.015	0.011	0.025	0.008
M5 Exemplar replay	0.008	0.024	0.016	0.117	0.029	0.159
M4 Prototypes, per-client	0.183	0.183	0.969	0.969	0.562	0.562
M4 Prototypes, aggregate	0.010	0.010	0.648	0.648	0.102	0.102
M8 Analytic, per-client	1.000	1.000	1.000	1.000	1.000	1.000
M8 Analytic, aggregate	0.993	0.993	1.000	1.000	0.997	0.997
M8 Analytic, state	0.911	0.816	1.000	1.000	0.981	0.956
Area under the ROC curve						
M0 FedAvg	0.559	0.563	0.657	0.622	0.678	0.693
M1 GLFC-style	0.555	0.608	0.616	0.779	0.645	0.778
M2 Gaussian replay	0.575	0.629	0.696	0.896	0.714	0.836
M3 FOT	0.570	0.603	0.680	0.732	0.691	0.729
M5 Exemplar replay	0.566	0.627	0.673	0.854	0.696	0.843
M4 Prototypes, per-client	0.883	0.883	1.000	1.000	0.977	0.977
M4 Prototypes, aggregate	0.681	0.681	0.989	0.989	0.842	0.842
M8 Analytic, per-client	1.000	1.000	1.000	1.000	1.000	1.000
M8 Analytic, aggregate	1.000	1.000	1.000	1.000	1.000	1.000
M8 Analytic, state	0.999	0.996	1.000	1.000	1.000	0.999

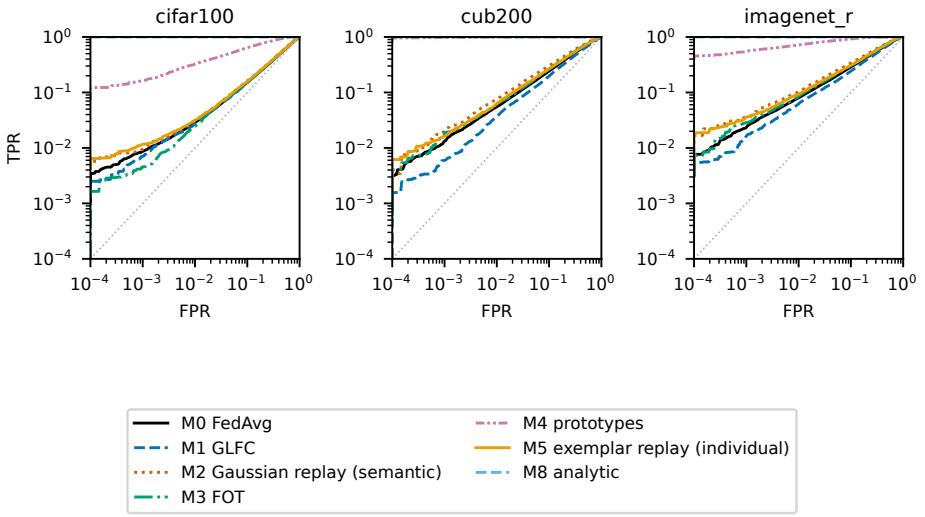


Figure 4: ROC curves of the membership attack on log scales, when the target task ends, for seed 0. For model updates the curve is the transcript attacker. For M4 and M8 it is the per-client release. The dotted diagonal is a random guess.

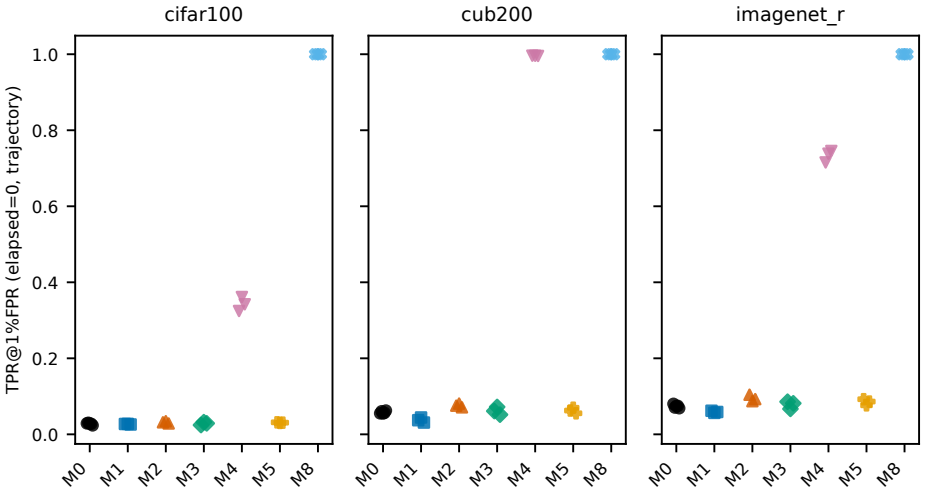


Figure 5: TPR at a 1% false positive rate when the target task ends, one marker per seed. For model updates the attack is the transcript attacker. For M4 and M8 it reads the per-client release. The spread across seeds is small compared with the differences between methods.

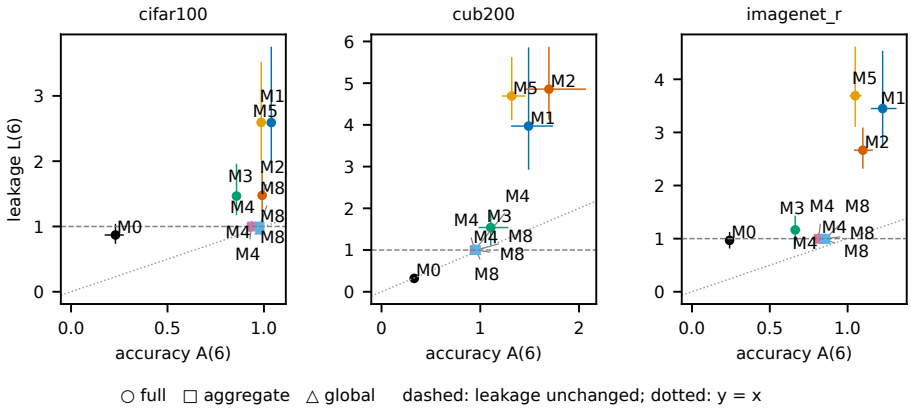


Figure 6: Accuracy kept, $A(6)$, against leakage kept, $L(6)$, six tasks after the target task. Circles are the transcript for model updates and the per-client release for M4 and M8. Squares are the aggregate. Triangles are the retained server state, labelled "global" in the legend. The dashed line marks unchanged leakage and the dotted line marks equal retention of accuracy and leakage. Bars are 95% intervals.

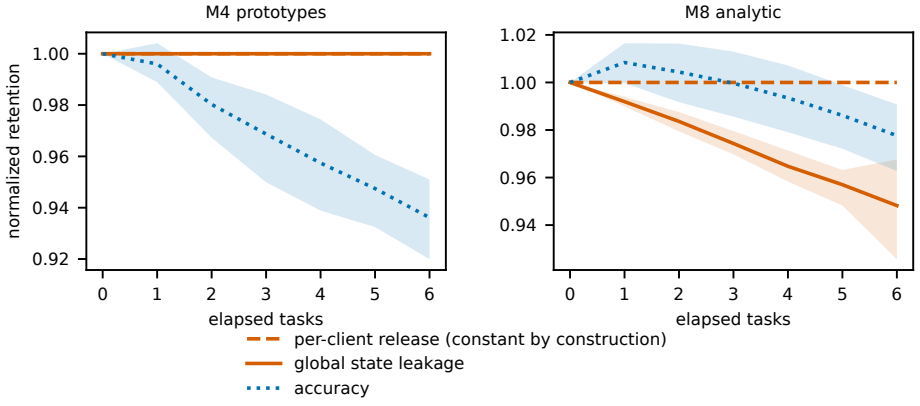


Figure 7: Retention of accuracy and leakage over elapsed tasks for the prototype method M4 and the analytic method M8 on CIFAR-100. The dashed line is the per-client release, which is constant by construction. The solid line is the leakage of the retained server state. The dotted line is accuracy. Bands are 95% intervals.

Figure 8 plots forgetting and leakage separately against the amount retained.

Table 12: Forgetting (−BWT) and TPR at a 1% false positive rate six tasks after the target task, on CIFAR-100, as the amount retained grows. The amount is the buffer size per class for M5 and the number of synthetic samples per class for M2. Brackets give 95% intervals over three seeds.

	M5 Exemplar replay		M2 Gaussian replay	
Per class	Forgetting	TPR	Forgetting	TPR
1	0.153 [0.125, 0.181]	0.038 [0.028, 0.047]	0.136 [0.109, 0.163]	0.025 [0.020, 0.029]
2	0.113 [0.091, 0.135]	0.047 [0.038, 0.056]	0.102 [0.091, 0.114]	0.030 [0.023, 0.036]
5	0.071 [0.058, 0.084]	0.056 [0.051, 0.061]	0.073 [0.062, 0.084]	0.034 [0.031, 0.037]
10	0.043 [0.035, 0.050]	0.062 [0.053, 0.070]	0.049 [0.038, 0.060]	0.037 [0.030, 0.044]
20	0.025 [0.022, 0.028]	0.075 [0.053, 0.098]	0.033 [0.025, 0.042]	0.043 [0.038, 0.047]
50	0.009 [0.001, 0.017]	0.083 [0.060, 0.107]	0.017 [0.008, 0.026]	0.046 [0.040, 0.052]

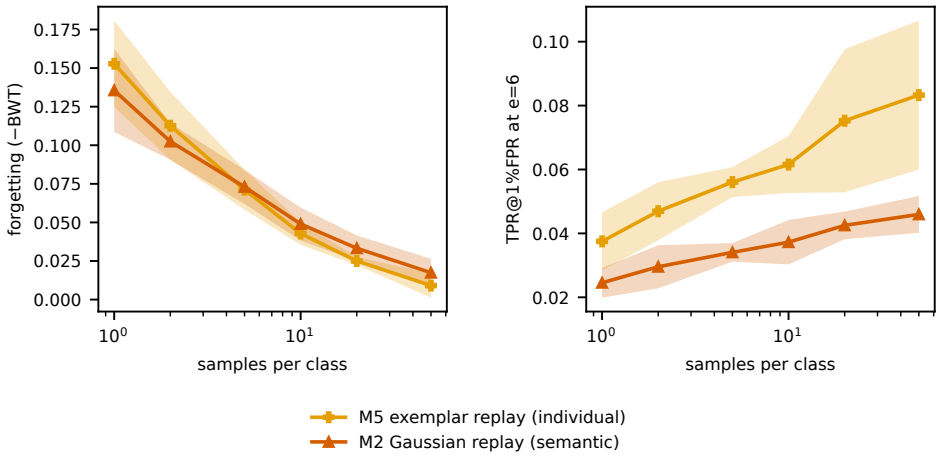


Figure 8: Forgetting (left) and TPR six tasks after the target task (right) against the number of samples retained per class, on CIFAR-100. Bands are 95% intervals over three seeds.

D.5 Secure aggregation on every dataset

Figure 9 shows the views of Table 3 when the target task ends and six tasks later, for four methods on each dataset.

D.6 Lifelong accounting over time

Table 13 gives the counts and costs behind Section **??**. M2, M3 and M4 release two records per task, which is why their count is 2.

Figure 10 extends Table 13 to 1,000 tasks for every method and unit. For our operator it shows the guarantee of Corollary 7 at $\epsilon_0 = 1$ and $\epsilon_0 = 4$.

D.7 Our operator

Table 14 repeats the accuracy of our operator with intervals. Table 15 gives the guarantee of our operator under each unit. It holds for any number of tasks under U_1 and U_2, as Theorem 6 states. Under U_3 with three tasks, $\epsilon_0 = 1$ becomes 2.36. Under U_4 it grows to 4.43 at ten tasks and 10.89 at fifty, as Corollary 7 requires.

Figure 11 plots the accuracy of Table 4 against ϵ_0. Figure 12 shows the full ROC curves of the audit in Table 4. Table 16 reports the effect of the decay γ, and Table 17 the effect of the number of clients under U_2.

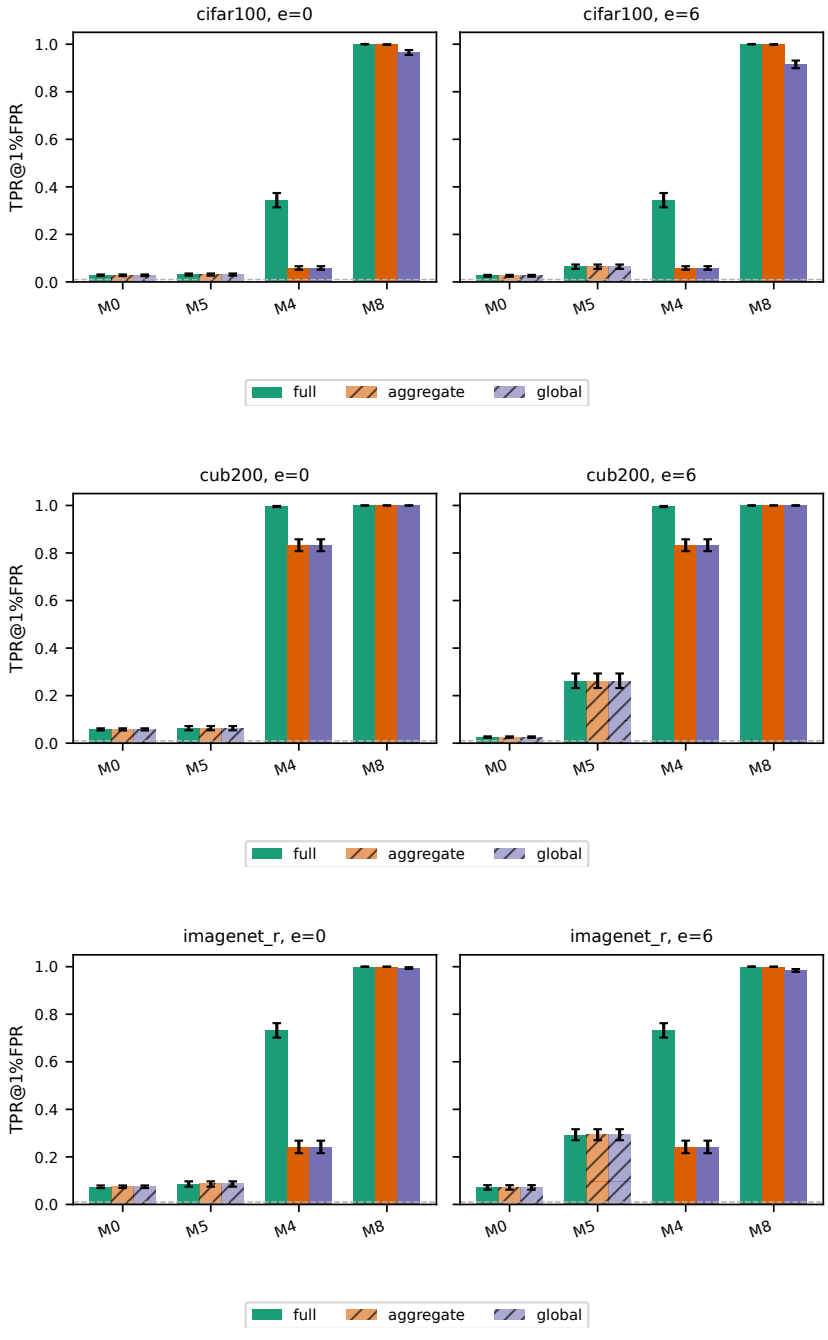


Figure 9: TPR at a 1% false positive rate by view, on CIFAR-100 (top), CUB-200-2011 (middle) and ImageNet-R (bottom). In the legend, "full" is the broadcast model for M0 and M5 and the per-client release for M4 and M8. "Aggregate" is what secure aggregation leaves visible. "Global" is the retained server state. For M0 and M5 the three views coincide by construction and are drawn hatched. Bars are 95% intervals.

Table 13: Lifelong accounting on CIFAR-100. The count m_{10} is the largest number of releases up to task t_{10} that read the data of one unit instance. ϵ is the cost of those releases at noise multiplier $z = 2$ and $\delta = 10^{-5}$ (Proposition 11). The last two columns extend the count to 1,000 tasks as described in Appendix C. U_3 uses a window of three tasks.

	Count m_{10}				ϵ at 10 tasks		ϵ at 1,000 tasks	
Method	U_1	U_2	U_3	U_4	U_1	U_4	U_1	U_4
M0 FedAvg	1	1	3	10	2.53	8.84	2.5	201.1
M1 GLFC-style	11	11	13	20	9.33	13.23	201.2	357.8
M2 Gaussian replay	2	2	6	20	3.64	13.23	3.6	357.8
M3 FOT	2	2	6	20	3.64	13.23	3.6	357.8
M4 Prototypes	2	2	6	20	3.64	13.23	3.6	357.8
M5 Exemplar replay	10	10	10	10	8.84	8.84	201.1	201.1
M8 Analytic	1	1	3	10	2.53	8.84	2.5	201.1

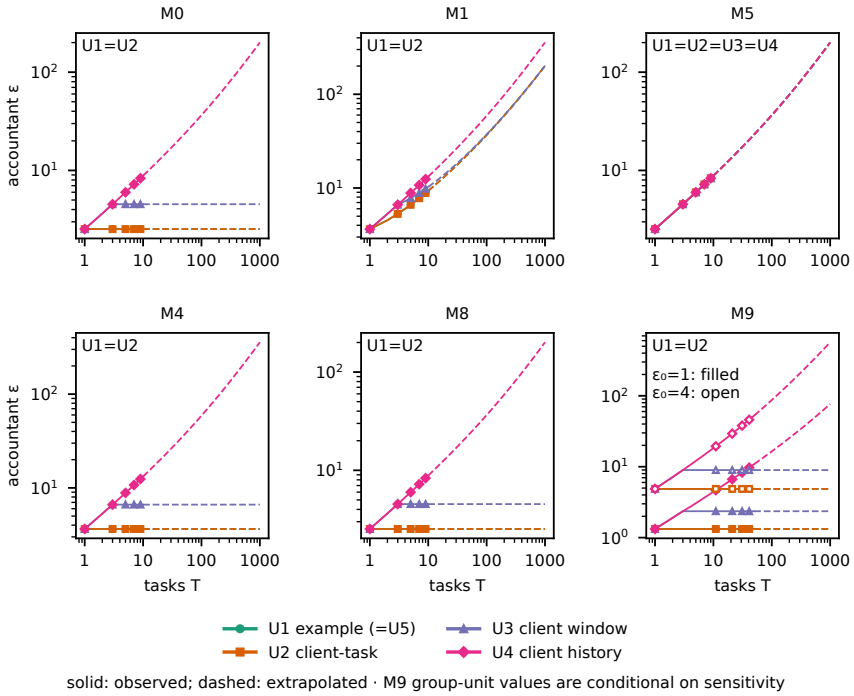


Figure 10: Accountant ϵ against the number of tasks, on CIFAR-100, at noise multiplier $z = 2$ and $\delta = 10^{-5}$. Solid lines are observed counts. Dashed lines extend them as described in Appendix C. In the legend, "U1 example" notes that U_5 equals U_1 in our benchmarks. The values of our operator (M9) under U_2 to U_4 require calibration to the client-level sensitivity.

Table 14: Final average accuracy of our operator after ten tasks with $\gamma = 1$, as in Table 4, with 95% intervals over three seeds.

	CIFAR-100		CUB-200-2011		ImageNet-R	
ϵ_0	U_1	U_2	U_1	U_2	U_1	U_2
0.5	0.366 [0.344, 0.387]	0.014 [0.000, 0.036]	0.008 [0.000, 0.016]	0.003 [0.001, 0.006]	0.015 [0.010, 0.020]	0.004 [0.000, 0.008]
1	0.596 [0.564, 0.628]	0.015 [0.000, 0.034]	0.017 [0.001, 0.034]	0.004 [0.000, 0.008]	0.082 [0.049, 0.115]	0.005 [0.003, 0.006]
2	0.690 [0.682, 0.697]	0.014 [0.000, 0.031]	0.050 [0.024, 0.077]	0.005 [0.000, 0.015]	0.203 [0.188, 0.217]	0.006 [0.005, 0.007]
4	0.810 [0.810, 0.811]	0.016 [0.003, 0.028]	0.088 [0.065, 0.112]	0.006 [0.000, 0.016]	0.324 [0.315, 0.333]	0.006 [0.002, 0.010]
8	0.818 [0.815, 0.821]	0.017 [0.011, 0.024]	0.161 [0.127, 0.196]	0.007 [0.000, 0.017]	0.385 [0.372, 0.398]	0.006 [0.004, 0.009]
∞	0.884 [0.883, 0.884]	0.880 [0.878, 0.883]	0.868 [0.868, 0.868]	0.848 [0.837, 0.858]	0.634 [0.633, 0.636]	0.624 [0.618, 0.629]
M8 Analytic, no privacy	0.883		0.871		0.635	
M0 FedAvg, no privacy	0.333		0.386		0.248	

Table 15: Guarantees of our operator on CIFAR-100 under each unit, for two values of ϵ_0. U_1 and U_2 follow from Theorem 6 for any number of tasks. U_3 (three tasks) and U_4 follow from Proposition 11 with $\delta = 10^{-5}$. The U_2 to U_4 values require calibration to the client-level sensitivity (Corollary 7).

	U_1	U_2	U_3	U_4	
ϵ_0	any k	any k	any k	$k = 10$	$k = 50$
1	1	1	2.36	4.43	10.89
4	4	4	8.97	18.31	52.77

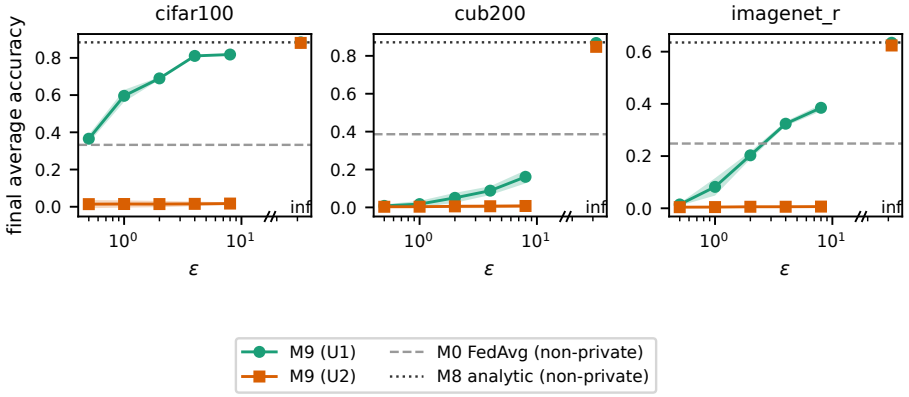


Figure 11: Final average accuracy of our operator (M9) against ϵ_0, with $\gamma = 1$. The point at the right edge of each panel is $\epsilon_0 = \infty$. The dashed and dotted lines are the non-private FedAvg and analytic method.

Table 16: Final average accuracy of our operator under U_1 for three values of the decay γ, averaged over three seeds.

	$\epsilon_0 = 4$			$\epsilon_0 = \infty$		
Dataset	$\gamma = 1$	0.9	0.7	$\gamma = 1$	0.9	0.7
CIFAR-100	0.810	0.713	0.371	0.884	0.867	0.735
CUB-200-2011	0.088	0.110	0.133	0.868	0.802	0.530
ImageNet-R	0.324	0.276	0.153	0.634	0.604	0.399

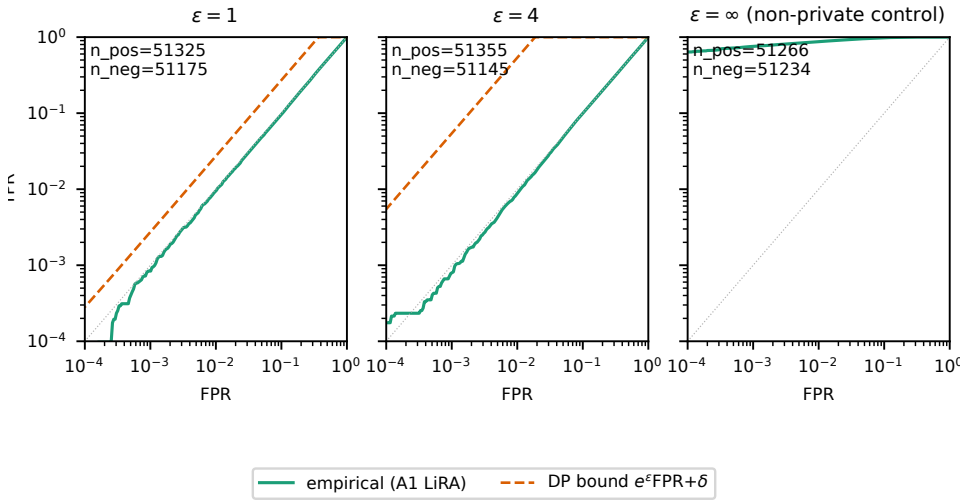


Figure 12: ROC curves of the membership audit of our operator on CIFAR-100 under U_1. The solid line is the empirical curve of the attack. The dashed line is the bound $e^{\epsilon}\,\mathrm{FPR} + \delta$ of Proposition 12. The right panel removes the noise and serves as a control.

Table 17: Final average accuracy of our operator under U_2 on CIFAR-100, with $\gamma = 1$, for three federation sizes. Averages over three seeds.

Federation	$\epsilon_0 = 1$	$\epsilon_0 = 4$	$\epsilon_0 = 8$
10 clients	0.015	0.016	0.017
50 clients	0.016	0.019	0.042
100 clients	0.016	0.018	0.048

Without noise, a smaller γ lowers accuracy on every dataset, because it gives the statistics of old tasks less weight. With noise at $\epsilon_0 = 4$, it also lowers accuracy on CIFAR-100 and ImageNet-R. On CUB-200-2011 it raises accuracy from 0.088 to 0.133. A smaller γ also shrinks the noise carried from earlier tasks, which may explain this difference. Under U_2, a larger federation helps only at $\epsilon_0 = 8$, and accuracy stays below 0.05.

D.8 Number of shadow federations

Our main M0 results use 4,096 shadow federations per seed. All other methods use 1,024. Table 18 compares the two budgets for M0 on CIFAR-100, with the same three seeds. The largest difference in TPR is 0.004, at six tasks after the target task.

Table 18: M0 on CIFAR-100 with 4,096 and with 1,024 shadow federations per seed, over the same three seeds.

	TPR at 1% false positive rate		Area under the curve	
Elapsed tasks	4,096	1,024	4,096	1,024
$e = 0$	0.029	0.029	0.560	0.554
$e = 6$	0.028	0.024	0.565	0.556

E Reconstruction from second-moment statistics

Section **??** shows that exact second-moment statistics reveal membership. Here we ask whether they also reveal the features themselves. In this appendix only, n denotes a number of examples, not of clients. We take n training examples of one class and form their statistic $X^\top X + \lambda I$ with $\lambda = 1$, where the rows of X are the feature vectors. The attacker removes λI and takes the top n eigenvectors and their scales. This determines the rows of X only up to an unknown rotation. The attacker resolves the rotation with public features of the same class from the reference set. It alternates two steps for 15 iterations: it matches each recovered vector to its most similar reference vector, and it solves for the best rotation given that match. We vary the quality of the reference data in three ways. The attacker uses the full reference pool of the class, a pool of 20 examples, or the full pool with added Gaussian noise of half the feature standard deviation. For each setting we run 25 trials and report the median over trials of the mean cosine similarity between recovered and true vectors. Recovered and true vectors are paired by position, not by best match.

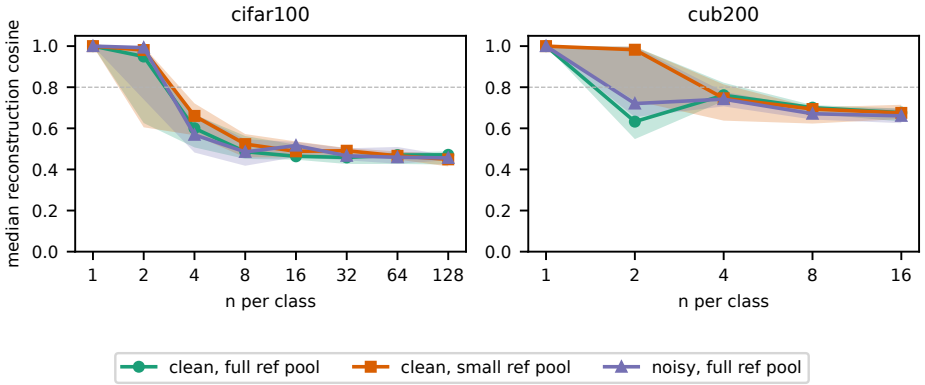


Figure 13: Reconstruction from a second-moment statistic of n examples of one class. The vertical axis is the median over 25 trials of the mean cosine similarity between recovered and true feature vectors. Bands are 95% bootstrap intervals. The three lines vary the quality of the public reference data.

Figure 13 shows the result. With one example the statistic reveals its feature vector exactly, with a cosine similarity of 1.000 on both datasets. This follows from the construction: a statistic of one

vector x is $xx^\top$, whose top eigenvector is x up to sign. With more examples the similarity falls. On CIFAR-100 it is 0.949 at $n = 2$, 0.600 at $n = 4$, and between 0.45 and 0.48 from $n = 16$ to $n = 128$. On CUB-200-2011 it is 0.761 at $n = 4$ and 0.676 at $n = 16$. For $n \geq 4$ the quality of the reference data changes these values little. We did not measure how similar unrelated examples of the same class are, and the pairing by position can understate recovery when $n \geq 2$. For $n \geq 4$ we therefore cannot say how much of the remaining similarity is recovery. The clear result concerns a single example. Its exact second-moment statistic reveals it completely.

F A NATURAL FEDERATION: CAMELYON17 PILOT

Our main task splits are synthetic. As a first check on a natural split, we ran M0 on Camelyon17 (Bandi et al., 2019), in its WILDS version (Koh et al., 2021). The data are histology patches from five hospitals, labelled as tumour or normal. We obtained the release from a public copy with the same schema and the same number of patches per hospital. We drew a subsample of 4,998 patches, stratified by class and hospital, and split it into 3,998 training, 500 test and 500 reference patches. Each hospital is one task, so all tasks share the same two classes. Each task has five clients. In the natural partition, a client holds all patches of a group of slides, and no slide is split between clients. In the comparison partition, the patches are divided by a Dirichlet split with parameter 0.5. Features, training and attack follow Section 5, with 1,024 shadow federations and three seeds.

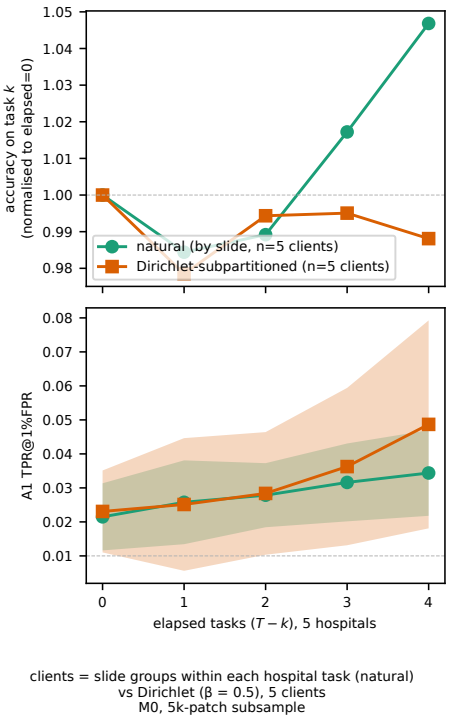


Figure 14: M0 on Camelyon17 with five hospital tasks and five clients per task. Top: accuracy on the target task relative to its value when the task ended. Bottom: TPR of the membership attack at a 1% false positive rate, with 95% intervals. Clients are formed by slide groups (natural) or by a Dirichlet split.

Figure 14 shows the result. When the target task ends, the TPR is 0.021 under the natural partition and 0.023 under the Dirichlet partition. Four tasks later it is 0.034, with an interval from 0.022 to 0.047, and 0.049, with an interval from 0.018 to 0.079. Accuracy on the target task changes little. It ends at 1.05 and 0.99 times its initial value. Because the hospitals share the same two classes, later tasks keep training the same classifier, and the pilot does not test forgetting. It shows that the audit runs on a natural partition and that leakage does not decay there within four tasks. The pilot is small, and its intervals are wide.

G Extended related work

Federated continual learning. FedWeIT (Yoon et al., 2021) splits each client model into shared parameters and sparse task-specific parameters, and lets clients reuse each other's task-specific knowledge. GLFC (Dong et al., 2022a) combines exemplar buffers on the clients with distillation from the previous global model. TARGET (Zhang et al., 2023) stores no exemplars. It trains a generator on the server and distils old tasks through synthetic data. FOT (Bakman et al., 2024) projects the updates of a new task away from the subspace that old tasks occupy. PILoRA (Guo et al., 2024) guides incremental low-rank adapters with class prototypes. Khademi Nori et al. (2025) combine latent exemplars with data-free techniques. AFCL (Tang et al., 2026) solves the classifier in closed form from second-moment statistics. Other work forgets biased knowledge on purpose (Wuerkaixi et al., 2024), manages memory in an online stream (Serra & Buettner, 2025), or builds on prompts and experts over frozen backbones (Wang et al., 2022b;a; Xu et al., 2025; Guo et al., 2026). Li et al. (2025) benchmark these methods under limits on storage, computation and labels. All of these papers measure accuracy and forgetting. Their privacy argument is that raw data stays on the client. To our knowledge, none of them measures what their retained artifacts reveal about old data as later tasks arrive. Our methods M1, M2, M3, M4, M5 and M8 keep the retention mechanism of six of these papers and test it.

Privacy attacks on federated and continual learning. Membership inference asks whether a record was part of the training data (Shokri et al., 2017). The likelihood-ratio attack of Carlini et al. (2022a) calibrates this test with shadow models, and we use it throughout. In federated learning, attacks read the updates of a single round. They infer properties of the training data (Melis et al., 2019), infer membership from white-box access (Nasr et al., 2019), or reconstruct inputs from gradients (Zhu et al., 2019; Geiping et al., 2020). A malicious server can make such reconstruction easier (Boenisch et al., 2023). These works study one round or one model. Time enters in two centralised studies. Jagielski et al. (2023) show that a model forgets examples seen early in training. Carlini et al. (2022b) show that removing the most exposed examples exposes others. Wu et al. (2024) measure membership and property inference against visual prompts, in a centralised setting with a single task. We study a different object: the federated transcript across a stream of tasks, read by an adversary who keeps every broadcast.

Differential privacy for streams and training. Differential privacy bounds what any release reveals about one unit (Dwork et al., 2006). Continual observation extends it to statistics released at every step of a stream (Dwork et al., 2010; Chan et al., 2011). Composition results bound the cost of many releases (**?**Mironov, 2017; Dong et al., 2022b), and the analytic Gaussian mechanism sets the noise exactly (Balle & Wang, 2018). For training, DP-SGD (Abadi et al., 2016) and its federated form with client-level units (McMahan et al., 2018) add noise to every update. Tree aggregation and matrix factorisation add correlated noise across rounds (Kairouz et al., 2021a; Denisov et al., 2022; Choquette-Choo et al., 2023). These mechanisms account for a training run in which each example contributes a known number of times. Our accountant asks the question that comes first in a continual setting. How many releases does the retention operator make read each datum, and does that number stay bounded as tasks arrive? Feldman & Zrnic (2021) also track privacy loss per record. Their filter stops using a record once its budget is spent. We use the per-record count for a different purpose, to classify retention operators. Lai et al. (2022) give a fixed privacy budget over a growing stream of tasks for a single centralised learner. We seek the same property in federation, where units and clients interact, and we show which retention operators can have it. Audits of private training compare the success of an attack with the guarantee (Steinke et al., 2023). Proposition 12 is the form of that comparison we use.